AMIGA WELT

3/88 Juli/August DM 12,–

ISSN 0933-8616

Alles über Drucker

- Langer Weg zum Standard
- IFF-Anim
- Spielehits auf 15 Seiten



Hallo Amerika

Datenreise in's Land der Telekommunikation

Super-listing

Icon-Manipulation komplett

Workbench 1.3

Ende der Spekulationen

sfr. 12,–; ÖS 95,–; Fmk 41,–; hfl 15,–; Lire 11 500;

Apropos ... Anregungen

Achtzig Prozent derer, die diese AmigaWelt in Händen halten, lesen so gut wie jeden Beitrag. So lautet eines der Ergebnisse unserer Leserumfrage vom April. Gute Noten erhielten „Format" und „Gesamteindruck". Mit Note 2,38 und 2,68 immer noch gut beziehungsweise befriedigend bewertet, stehen „Titelbild" (Max Headroom) und „Umfang der Anzeigen" am Ende der Bewertungsskala. 46 Prozent unserer Leser besitzen einen Amiga 500, 15 Prozent den Amiga 1000 und beachtliche 23 Prozent den Amiga 2000. Sie werden in zwei Drittel aller Fälle ausschließlich privat und von einem guten Drittel unserer Leser sowohl privat als auch beruflich genutzt. All diese Zahlen und Fakten sollen uns Anregungen geben, um die AmigaWelt noch besser gestalten zu können. Als Dankeschön für Ihre Bemühungen verlosten wir unter den Einsendern des Fragebogens einen Fujitsu DX 2100, einen Qualitätsdrukker mit 9-Nadel-Technologie, besonders geeignet für Textverarbeitung oder Grafik, womit sich laut Umfrage 60 beziehungsweise 70 Prozent unserer Leser beschäftigen. Der Drucker wurde von der Macrotron AG zur Verfügung gestellt. Und wer hat gewonnen? Gewonnen hat Herr J. Leis aus 6730 Neustadt.
Herzlichen Glückwunsch!
Um Anregungen und Informationen ging es auch Michael Nickles. Für die MS-DOS-Welt und für die AmigaWelt besuchte er die Frühlings-COMDEX in Atlanta. Sein Urteil: „Viel Neues für Amiga." So zog die auf den Bedarf von Wiederverkäufern zugeschnittene Messe mit rund 60 000 Besuchern 20 Prozent mehr Interessenten an als im Vorjahr. Die Händler allerdings erwarteten von der Frühjahrs-Ausgabe der mittlerweile größten amerikanischen Computermesse weniger sensationelle Produkt-Neuheiten und freuten sich vor allen Dingen über die lokkere Atmosphäre. Ein Computerhändler aus North Olmstead in Ohio: „Meine Leute haben von der Messe nicht groß profitiert. Aber sie sollten mal aus dem Alltagsgeschäft hinaus kommen und ein bißchen Spaß haben." Glenn Paul aus Cranbury, New Jersey, kam zu der Auffassung, daß die Ausstellung eine gute Gelegenheit sei, um Kontakte aufzufrischen und neue zu schließen: „Die Frühlingsausstellung ist viel kleiner als ihre Schwester in Las Vegas, deshalb ist die Atmosphäre gemütlicher." Genau das machen einige Eisenherzen der COMDEX in Atlanta zum Vorwurf. „Las Vegas is the real thing", meinte Tom Jacobs, ein Computerhändler aus Nashua, New Hampshire. „Dort hat die Sache angefangen, dort finden die besseren Ankündigungen statt." Wir werden sehen.



Redaktionsassistentin Siggi Pöschel zieht den Briefumschlag von J. Leis aus 6730 Neustadt. Er hat einen DX 2100 im Wert von 1500 Mark gewonnen.

Manfred S. Schmidt

Manfred S. Schmidt

INHALT

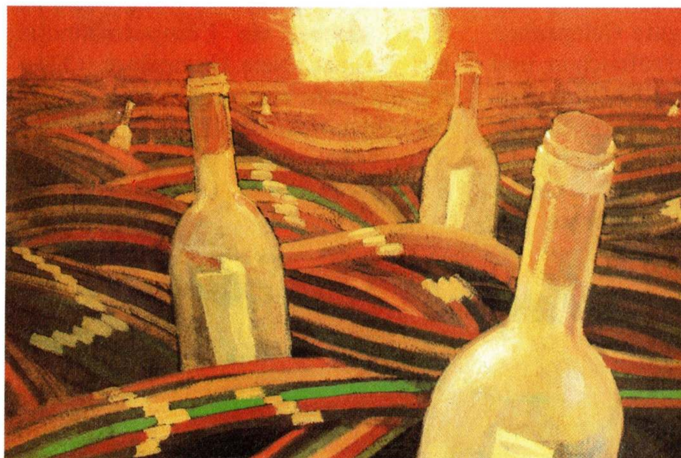


Wenn sich zum Computer ein Drucker gesellen soll, wird es Zeit, sich mit dessen Grundlagen zu beschäftigen.

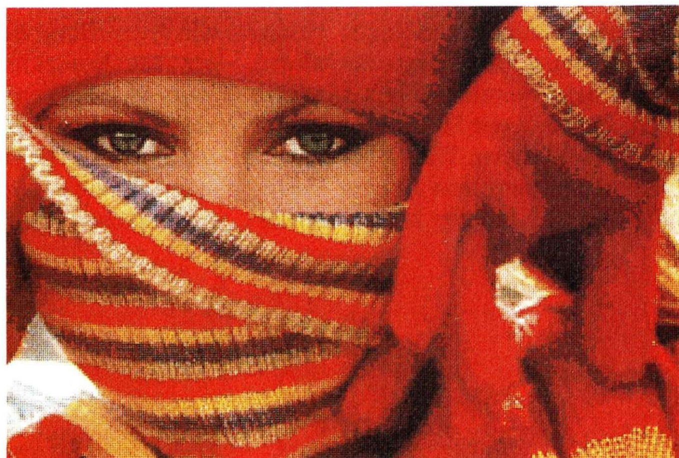
Notepad

Amigawelt

Hard- und Software



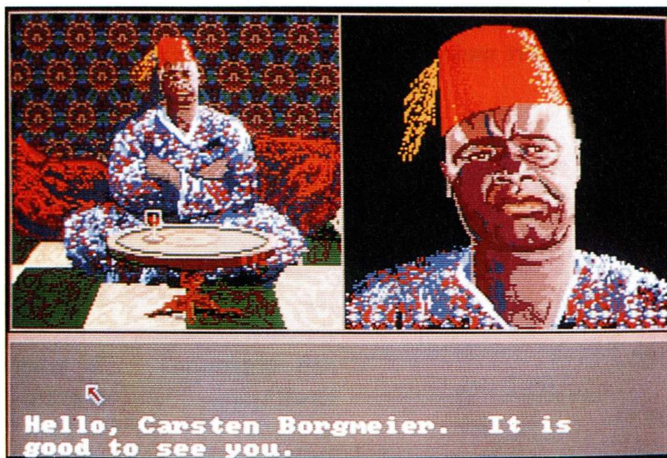
Sehr viel schneller als Flaschenpost: mit DFÜ amerikanische Informationsquellen anzapfen **Seite 25**



Wie auf dem Bildschirm, so auf dem Papier: das ist der Anspruch an Farbdrucker **Seite 66/73**



In der Computermalschule erklärt Joel Hagen seine grafische Fiktion vom außerirdischen Leben Seite 90



„Return to Atlantis“ vermittelt exotische Bekanntschaften und führt in geheimnisvolle Tiefen Seite 129

Ferien mit Amiga

Lagerfeuerromantik und sportliche Aktivitäten mit der Faszination von Bits und Bytes verbinden, lautet die grundlegende Idee der Veranstalter von Computerferien. Etablierte Camps belegen, daß diese Mixtur aus scheinbar gegensätzlichen Elementen erfolgreich ist.



Computerbegeisterte unter sich. Da wird Wissen ausgetauscht, Trickreiches vorgeführt und ausprobiert.

„CompuCamp" bietet während der Herbstferien '88 Amiga-Kurse an. Im holsteinischen Westensee können sich Anfänger mit Workbench, Preferences, Benutzeroberfläche und AmigaBasic vertraut machen. Fortgeschrittene steigen ein in Amiga-DOS, lernen sowohl die Workbench zu manipulieren, Basic-Programme zu erstellen, Assembler-Routinen einzubinden als auch Grafik- und Soundprogrammierung. Für den körperlichen Ausgleich bietet der Veranstaltungsort Tennis und American Sports. Eine Woche Unterbringung, Verpflegung und Computerkurs kosten zusammen 675 Mark. „Computerworld" arbeitet im Freiburger Camp ebenfalls mit Commodore Amigas. Wochenkurse finden zwischen dem 2. Juli und 20. August statt. Urlauber lernen, mit dem Betriebssystem umzugehen und mit Amiga-spezifischen Programmen zu arbeiten. Eine Woche Vollpension im Jugendgästehaus plus Amiga-Kurs und einiger Freizeitaktivitäten kosten 610 Mark. Für Sportkurse und Ausflüge zahlt man zusätzlich 80 Mark.

Wen die Berge nicht stören, der kann in Österreich Computerferien machen. „Young Austria" bringt seine Gäste im Jugendferienhof Gallend bei Bad Gastein unter. Die nächsten Drei-Wochen-Kurse beginnen am 30. Juni, 21. Juli und 11. August. Auf PC und neuen Commodore-Rechnern lernen Anfänger das Betriebssystem MS-DOS bzw. CP/M kennen. Anfänger programmieren in Basic, Fortgeschrittene in Pascal oder Modula II. Für einen kompletten Camp-Urlaub zahlt man 1320 Mark.

Im „Ferienclub Lüneburger Heide" werden im Sommer Kurse zwischen dem 18. Juni und Ende August, im Herbst zwischen dem 2. und 30. Oktober angeboten. Gearbeitet wird noch mit dem C64. Im Sommer sollen sie den Amigas weichen. Einsteiger und Fortgeschrittene befassen sich drei Stunden täglich mit Betriebssystem, Basic und Assembler. Am Nachmittag geht's auf den Tennisplatz oder auf die Reitbahn. Gruppen residieren im Clubhaus, Familien in Ferienwohnungen. Eine Woche kostet 490 Mark.

Alle genannten Kurse sprechen im wesentlichen Jugendliche zwischen Zehn und Zwanzig an. Dennoch sind erwachsene Freaks und Neulinge willkommen. Wer sich für Fortbildungs- und Kreativ-Urlaub interessiert, fordert unter folgenden Adressen und Telefonnummern Informationsbroschüren an:

CompuCamp,
Tel. 0 40/81 10 81;
Ferienclub Lüneburger Heide,
Tel: 0 95 62/6 09;
Young Austria,
Tel. 00 43/662/25 75 80;
Computer World,
Tel. 07 61/4 47 75.

(ub)

Computermesse zwischen Ost und West: IFABO '88 in Wien

Für ausländische Unternehmens- und Firmengruppen wird der österreichische Markt als Drehscheibe zwischen Ost und West immer interessanter. Das bewies die internationale Beteiligung an der IFABO vom 3. bis 7. Mai auf dem Wiener Messegelände.

Mit 528 Direktausstellern (davon 36 aus dem Ausland), die insgesamt 1233 Firmen (98 österreichische, 1135 ausländische aus 28 Ländern) vertreten, auf 43 708,5 Quadratmetern Ausstellungsfläche und 88 964 Besuchern war die diesjährige IFABO die erfolgreichste überhaupt. Sie ist damit nicht nur zur größten österreichischen Fachmesse, sondern auch zu einer der größten Europas geworden. Diese Expansion schlug sich voll auf die integrierte Software-Messe PROGRAMMA nieder, die zum fünften Mal abgehalten wurde. Erstmals nicht mehr auf der Galerie 15 als Anhängsel, sondern in zwei riesigen Hallen, um den 152 Direktausstellern die erforderlichen 4500 Quadratmeter bieten zu können. Die Trennung in Anwendungen für PCs und Mikros in Halle 1 und der Ausstellung der Systemhäuser, Rechenzentren und Groß-EDV-Softwarehäuser in Halle 12 kanalisierten die Besucherströme in die richtigen Wege. Obwohl branchenorientierte Standardanwendungen eindeutig im Mittelpunkt des Interesses standen, wurde bei CAD/CAM noch mehr gezieltes Kaufinteresse bemerkt. DTP wurde in weiterentwickelten Formen präsentiert. Schon auf der CeBIT zeichneten sich neue Trends ab, die auf der IFABO in Wien deutlicher wurden: Zuerst war die Hardware, dann kam die Software und jetzt wird auch die Brainware wichtig. Gerade bei EDV-Herstellern und -Anwendern wächst der Bedarf an spezialisierten Mitarbeitern. Ein Beweis dafür ist, daß allein in den österreichischen Tageszeitungen und in der COMPUTERWELT-Österreich im Jahr 1987 rund 40 Millionen Schilling für EDV-Stelleninserate ausgegeben wurden.

Ein Beitrag zur Bewältigung der Personaldefizite war daher eine Initiative, die von CW-Publikationen Österreich gemeinsam mit der Wiener Messe unter der Schirmherrschaft von Bundesminister Alfred Dallinger im Rahmen der IFABO getestet wurde: Der Karriere-Treffpunkt. Es war das eine arbeitsmarktpolitische Initiative mit dem Zweck des persönlichen Kontaktes von Jobsuchenden und den personalsuchenden Unternehmen. Neben der Präsentation von Karrieremöglichkeiten brachte der Karrieretreffpunkt auch ein umfangreiches Programm an Vorträgen über Berufsaussichten in einer immer bedeutender werdenden Branche. Der Karriere-Treffpunkt wurde ein voller Erfolg. Die daran teilnehmenden Firmen lobten eine hohe Frequenz sowie ein wirkliches Interesse an einer neutralen Beratung für EDV-Berufe.

Dominiert war die IFABO natürlich vom Computer und ist damit zur größten Computer-Messe in Österreich geworden. Die PC sind noch immer Spitzenreiter.

So kurz nach der CeBIT gab es natürlich so gut wie keine Hardware-Neuheiten. Es konnten lediglich schon konkretere Liefertermine genannt werden. Die Entscheidungskriterien für den Kauf sind aber hier extrem von den dafür vorhandenen Softwarelösungen abhängig. Demzufolge war auch das Gedränge in den PROGRAMMA-Hallen beängstigend.

Die Präsentation von Commodore folgte diesen neuen Trends und zeigte eindeutig, wohin Commodore in den nächsten Monaten und Jahrene marschieren wird. In einem Pressegespräch brachte das Harald Speyer etwa wie folgt zum Ausdruck: Der Weg geht eindeutig aufwärts, und das nicht nur im Home-Computer-Bereich, sondern auch im professionellen Sektor. Einen wichtigen Platz nimmt hier zweifelsohne auch die ganze Amiga-Palette ein. Für uns Europäer war es erfreulich zu hören, daß Commodore derzeit zwei Drittel des Umsatzes in Europa macht. Die durch Harald Speyer bekanntgegebenen Zahlen sprechen für sich. Die Zahlen waren so brandneu, daß sie Speyer von einem Schmierzettel ablesen mußte. Doch bereits am Ende des Pressegespräches standen sie bereits als gedrucktes Extrablatt zur Verfügung. Ein Amiga 2000 mit „Pro-



CW-Österreich nahm sich der Nachwuchssorgen der Branche an.

EXTRABLATT
COMMODORE STANDINFORMATION IFABO'88

Condensed Consolidated Statements of Operations
Commodore International Limited and Subsidiaries
(In Thousands of Dollars, Except Per Share Amounts)

	For the Three Month Ended March 31			For the Nine Month Ended March 31		
	1988	1987		1988	1987	
Net Sales	200.3	165.5	+ 18%	656	616	+ 6%
Net Income	9.6	1.0	+ 860%	43.6	26.5	+ 65%

Mit Professional Page, Laserdrucker und Amiga sofort verfügbar: Extrablatt von Commodore.

fessional Page" und ein „AST Turbo-Laser/PS" machte das möglich.

Dieses System lieferte auch eine bunte Überraschung: Nicht nur die Blätter der Standzeitung, sondern auch der Farbumschlag wurde damit gestaltet. Die Eingabe der Farbbilder erfolgte über Video-Kamera mit Echtzeit-Digitizer oder Scanner. Die Texte wurden mit Wordperfect erfaßt oder direkt ins Layout geschrieben. Als Scanner stand der von der Print-Technik vertriebene „Silver Reed CP11" zur Verfügung. Dieser tastet Dokumente mit einer Auflösung von 1760 × 2048 Punkten ab. Die Bildschirmdarstellung umfaßt drei Modi, die auf einem Hires-Interlace-Screen mit 640 × 512 Punkten darstellbar sind.

Automatisch produzierte der Amiga die vier druckgerechten Farbauszüge der am Bildschirm layouteten Seiten und speicherte sie im Postscript-Format auf Diskette ab. In der Druckerei wurden dann direkt von dieser Diskette ohne weitere Bearbeitung die Filme für den Vierfarben-Druck belichtet — innerhalb einer guten Stunde! Damit erspart der Amiga viel Zeit und Geld. Erstmals können Vierfarbprospekte völlig frei und ohne Belastung durch hohe Lithokosten gestaltet werden. Am Bildschirm ist das fertige Produkt 1:1 zu sehen. Umgestaltungen, Ausschnitte und Formate von Vierfarb-Bildern zu ändern, kostet nur mehr einen Mausklick.

Diese Fähigkeiten können nicht nur für die Produktion von Druckwerken eingesetzt werden. Mit Desktop-Presentation eröffnet der Amiga in nahezu jedem Bereich der Informationsvermittlung neue Perspektiven. Präsentationsmappen, Overheadfolien für Vorträge und vieles andere können jetzt ohne großen Aufwand perfekt gestaltet werden.

Amiga dringt in den Bürobereich vor. So zeigte das Studio Helmut Kratky (1190 Wien) am Commodore-Stand das Buchhaltungspaket PROFIBU. Das Bemerkenswerte daran ist, daß es am 2000er ohne PC-Karte und daher natürlich auch am 500er läuft. Für 150 Konten genügen 512 KByte. Steht 1 MByte zur Verfügung, dann können bereits 800 Konten verwaltet werden. Dazu gibt es ein in WYSWYG verfaßtes Handbuch, mit dem auch Computer-Laien problemlos zurechtkommen.



Amiga-Arbeitsplatz mit X-Cad, Pronet und Proboard.

Der Amiga dringt bereits in Bereiche ein, die bisher professionellen CAD-Workstations vorbehalten werden. Beispiele dafür sind X-CAD, PRO-NET und PRO-BOARD.

X-CAD ist benutzerfreundlich und leistungsstark. PAL und deutsche Umlaute werden unterstützt. Eingaben können über die Tastatur, mit der Maus und über Grafiktablett erfolgen. Die Bildaufbauzeit ist erstaunlich schnell. In acht verschiedenen Tiefen sind 255 Layers übereinander möglich. Plotter, Laserdrucker und Epson-kompatible Matrixdrukker werden unterstützt. Mögliche Speicherformate: IFF, X-CAD und als Plot-File für einen Hintergrund-Task.

PRO-NET dient zum Entwickeln von Schaltungen, die dann PRO-BOARD entflechtet und reprofähige (auch Multi-Layer) Vorlagen druckt.

Ein auf der CeBIT noch nicht präsentiertes Paket ist der Amiga 2000 E, der flexible Schulcomputer. Wichtig für die Schulverwaltungen ist, daß der Amiga in einem Netzwerk betrieben werden kann und durch die offene Systemarchitektur zukunftssicher ist. Schulen, Schüler und Studenten können mit entsprechenden Bescheinigungen in den Genuß dieses preisgünstigen Amiga 2000 E gelangen. Ansonsten stand die diesjährige IFABO-Präsenz von Commodore-Österreich im Zeichen des Jubiläums „10 Jahre PC". Es wurde eine PC-Palette gezeigt, die vom industriekompatiblen Einsteigermodell bis zum 386er Modell alle Anwendungsgebiete abdeckt.

Auf der PROGRAMMA war der Amiga nur mit einigen grafischen Anwendungen zu sehen. Für den Bürobereich war die Firma Mesonic ein „weißer" Rabe und präsentierte eine Vorab-Version von IRIS-LOHN, ein auf österreichische Verhältnisse zugeschnittenes Lohnverrechnungs-Programm.

Das Programm ist in C geschrieben und ist eine Umsetzung der in Österreich gut bekannten PC-Version gleichen Namens. Einige Software-Häuser arbeiten an Umsetzungen von PC-Standardpaketen auf den Amiga. Hier ist für die nächste IFABO einiges zu erwarten.
(R. Wolf)

AMIGA tele VISION beim Reitturnier

Commodore sponsort Sport. Das ist nicht neu. Neu aber ist das rollende Amiga-Medienzentrum. Acht Amigas helfen Sportkommentatoren, verschiedene Informationskanäle zu nutzen und Material aufzubereiten

„Aufbau von Image und Bekanntheitsgrad gegenüber Sportinteressierten sind Basis eines speziellen Marketingkonzeptes", betonte Winfried Hoffmann, Devision-Vice-President Marketing und Vertrieb von Commodore Europa, anläßlich einer Pressekonferenz in Wiesbaden. Im Rahmen des traditionellen Wiesbadener-Reitturniers stellte Commodore sein neues Mediamobil „AMIGA tele VISION" vor. Das rollende Rechenzentrum und Videostudio ist mit allen modernen, digitalen Trickeinrichtungen ausgerüstet. Durch 17 eingebaute Computer und modernste Videosysteme erhält die Aufbereitung von Sportergebnissen als Service für die Fernsehanstalten und die schreibende Presse eine neue Dimension. Das zehn Tonnen schwere Mediamobil wurde beim Wiesbadener Reitturnier für die Auswertung sowie zur Vorbereitung von Einblendungen in die Berichterstattung eingesetzt. Über die Computersysteme wurden nicht nur die Bildinformationen aufbereitet und zum Teil grafisch umgesetzt. Auch Überblendungen und Mischungen wurden vom Computer gesteuert. Damit haben die beiden Welten, Computer und Video zum ersten Mal so unmittelbar über elektronische Schnittstellen miteinander gearbeitet. Das Fahrzeug ist für den Einsatz bei Großveranstaltungen im europäischen Raum zugeschnitten. Neben festen vertraglichen Vereinbarungen besteht darüber hinaus die Möglichkeit, daß Fernsehanstalten das Fahrzeug für besondere Einsätze mieten können. Die Computerwelt des „AMIGA tele VISION" umfaßt 17 Mikrocomputer. Davon sind neun Commodore AT 40/40 Rechner mit dem 80286-Prozessor. Diese Systeme sind jeweils in Dreiergruppen senkrecht in Tischen eingebaut, um Platz zu sparen. Dazu kommen acht Amiga 2000 in Vollausstattung. Mit Festplatten und Diskettenlaufwerken.

Am SK-Netz hängen 17 Computer

Drei davon sind GEN-Lock-fähig, zwei sind mit Video-Digitizer ausgestattet. Alle Computer sind auf drei Netzwerkebenen verbunden.
An dem SK-Netz hängen alle 17 Maschinen. Dazu gibt es ein Hardware-adressierbares V.24-Netz. Es bietet den Vorteil, daß jedes Gerät mit einer Nummer versehen und so direkt ansprechbar ist. Man kann also zum Beispiel den Laserdrucker unter Nummer 15 von jedem Rechner aus erreichen. Rückgrat der Kommunikation untereinander sind die Schnittstellen. In jedem AT sind zehn serielle Schnittstellen eingebaut. Damit die Verbindung auch klappt, wurde ein spezielles Schnittstellensteckfeld eingebaut, wo alle Geräte beliebig, je nach Bedarf miteinander verbunden werden. Im Mediamobil sind 210 serielle Schnittstellen installiert, von denen derzeit 184 in Betrieb sind. Dabei handelt es sich nicht nur um ein serielles Computer-Schnittstellen-Konzept, sondern gleichzeitig um die Brücke zur Fernsehebene. Denn über die in den Fernsehsystemen ebenfalls eingebauten seriellen Schnittstellen ist das Konzept der Computersteuerung des Fernsehbereichs erst realisierbar.

Die Fernsehwelt besteht aus 2 U-matic-Highband-Recordern sowie einer C-Bandmaschine. Dazu kommen ein Bildmischer und ein Tonmischpult mit 10 Kanälen. Der technologische Vorsprung im Fernsehbereich besteht jedoch in den 5 kombinierten Schrift- und Grafikgeneratoren, die durch ein spezielles digitales Effektgerät beliebig miteinander kombiniert werden können. Dieses Effektgerät mit dem Namen Pinacle basiert auf einem ganz normalen 80286-Rechner, in dem eine ähnliche Philosophie verwirklicht wurde, wie sie in der offenen Systemarchitektur der Amiga-Baureihe besteht. Mit Hilfe von Brükkenkarten steuert der MS-DOS-Rechner auf dem Video-Bus ganz bestimmte, im Fernsehen wichtige Anwendungen. Das reicht von digitalen Effekten wie Ver-



10 Tonnen modernster Computer- und Video-Technologie setzt Commodore bei Sportveranstaltungen für Auswertung und zur Unterstützung von Fernsehübertragungen ein: ein komplettes Rechenzentrum und Video-Studio auf Rädern. ▶

AMIGA WELT



AMIGA WELT 4/88 Ab 22. August im Zeitschriftenhandel — oder beim Verlag bestellen!

Bereits erschienen:

AMIGA WELT 3/88

AMIGA WELT 2/88
Baseball auf dem Bildschirm · Neue Grafikchips, Transputer, Amiga-Unix, AT-Karte, Hires-Monitor Grafik in Bits und Bytes · Einmaleins der Maschinensprache · Grafikanimation unter DPaint

AMIGA WELT 1/88
Computer-Kreatur Max Headroom · Grafik-Kurs: Computeratelier für jedermann · Listings · Spiele-Hits auf 15 Seiten

AMIGA WELT 3/87
Juggler-Demo im Eigenbau · Butcher · Deluxevideo 2 · Programmiersprachen: C-Kurs/Turbo C/ Turbo Prolog · DTP unter MS-DOS

AMIGA WELT 2/87
Faszination in Noten · Desktop Publishing · Genlock · Video-Clips selbstgemacht · MS-DOS · Listings · Systemroutinen · Tips und Tricks



CW PUBLIKATIONEN
Ein Unternehmen der IDG Communications
Ihre starke Verbindung zur Computerwelt

Bitte ausgefüllten Bestell-Coupon abtrennen und einsenden an: CW-Publikationen Verlagsgesellschaft mbH, Vertrieb, Postfach 40 04 29, D-8000 München 40.

B e s t e l l - C o u p o n

JA, ich bestelle aus Ihrem Angebot folgende(n) Titel:

_ Exemplar(e) AMIGA WELT 4/88
_ Exemplar(e) AMIGA WELT 3/88
_ Exemplar(e) AMIGA WELT 2/88
_ Exemplar(e) AMIGA WELT 1/88
_ Exemplar(e) AMIGA WELT 3/87
_ Exemplar(e) AMIGA WELT 2/87

zum Einzelpreis von jeweils DM 12,—.

Gewünschte Zahlungsweise bitte ankreuzen:*
☐ Per Verrechnungsscheck (liegt bei) (Heftpreis(e) + DM 2,— Porto**)
☐ Per Nachnahme (Heftpreis(e) + DM 2,— Porto** + DM 1,70 Nachnahmegebühr)

*Auslandsbestellungen nur gegen Verrechnungsscheck
**Ab 2 Exemplaren erhöht sich der Portoanteil auf DM 3,—

Datum/Unterschrift

Firma (falls Lieferanschrift)
Name
Vorname
Straße, Hausnr./Postfach
PLZ Ort

S78

70 000 Mark für die Kreativsten

Einsendeschluß 1.9.88 !!!!

Auf geht's zum Endspurt! Wer seine Ideen zu einem Amigaspiel noch immer nicht zu Papier oder Diskette gebracht hat, hat noch Chancen. Bis zum 1. 9. 88 (Poststempel gilt) müssen Konzept (Storybook und Grafiken) beziehungsweise fix und fertig ausgearbeitetes Programm an die Redaktion geschickt worden sein.
Erster Preis beim Konzeptwettbewerb ist ein Geldbündel von 3000 Mark. Das beste ausgearbeitete Programm wird mit 40 000 Mark belohnt. Die Vergabe des Hauptgewinns ist an bestimmte Kriterien geknüpft: Dem Programm muß eine hervorragende Spielidee zugrunde liegen, die Realisation muß erstklassig und das Spiel kommerziell verwertbar sein.
Wer mitmachen will, sendet seinen Beitrag exklusiv, nebst Angaben zur Person, plus Programmbeschreibung und Abtretungserklärung über die Nutzungsrechte an:

CW-Publikationen
Redaktion AmigaWelt
Kennwort „Amiga-Wettbewerb"
Postfach 400 429
8000 München 40

Programmierwettbewerb Amigaspiel

- **1. Preis**
  40 000 Mark
- **2. Preis**
  10 000 Mark und eine Reise zu Cinemaware/USA
- **3. Preis**
  Amiga 2000 mit AT-Karte und Harddisk
- **4. Preis**
  Listing des Monats — 2000 Mark

Konzeptwettbewerb

- **1. Preis**
  3000 Mark
- **2. Preis**
  2000 Mark
- **3. Preis**
  1000 Mark

Einsendeschluß: 1.9.88

Bitte Abschnitt unterschreiben, ausschneiden und mitschicken!
Der Rechtsweg ist ausgeschlossen

Wenn ich zu den glücklichen Gewinnern gehöre, dürft Ihr von der AmigaWelt — CW-Publikationen, München, und Time Warp Productions, Gütersloh, mein Programm benutzen.

Unterschrift



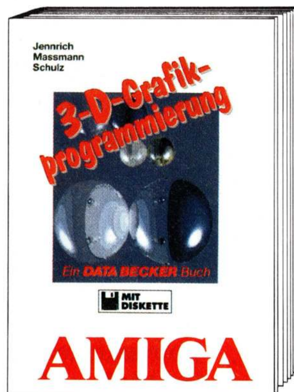
größern und Verkleinern über digitale Speicherung von Standbildern, die auf der Festplatte abgelegt werden, und die sogenannte Paint-Box bis zur Entwicklung von Vollbildern und Spezialbereichen wie Computeranimation und dreidimensionalen Trick-Effekten. Es handelt sich hierbei um ein völlig neues, sehr modulares Konzept.
Hier wird Computertechnik auf sehr spezielle Art und Weise zur Realisierung von Videoprojekten nutzbar gemacht. Der Nutzeffekt für die Fernsehanstalten ist, daß sie Bildinformationen in der gewünschten Form erhalten. Das könnten einerseits Schriften als Einblendungen in das laufende Bild sein. Das können auch komplett gestaltete Vollbilder sein, die wie eine Extra-Kamera verstanden werden. Mit Hilfe des im Commodore-Mediamobil eingebauten Equipments werden zum Beispiel Informationen aus drei Schrift- und Grafikgeneratoren über das Pinacle-System zu einem Signal addiert. Hier hat sich eindeutig die Computertechnik auf den Bedarf des Fernsehens eingestellt.
Ein weiterer Nutzen ist die Speichermöglichkeit von bis zu 200 Videostandbildern. Hier werden aus dem bestehenden Bandmaterial ideale Szenen herausgelöst und wie ein Einzelbild abgespeichert. Sie sind dann auf Anforderung sofort abrufbar und in bestehende Vollbilder einblendbar.
Während der AT-Bereich der Computerwelt im Mediamobil zur Steuerung der Videoanlage verwendet wird, sind die acht Amiga 2000 dem Präsentationsbereich zuzuordnen. Sie werden verwendet, um zum Beispiel Kommentatoren verschiedene Informationskanäle anzubieten und Materialien aufzubereiten. Die in den Amigas vorhandenen Bilder wie zum Beispiel Logos, Schriften oder ähnliches sind beliebig abrufbar. Die Amigas werden nicht handbedient, sondern stehen voll unter Kontrolle der anderen Rechner.
Das Herzstück des Mediamobils ist das Steckfeld. Alle Monitore im Auto sind vollkommen frei belegbar. Damit kann das Fahrzeug jeder Anwendung, jeder einzelnen Sportveranstaltung beliebig angepaßt werden. Hier besteht zu standardmäßig ausgestatteten Übertragungswagen mit ihren festen Produktionsabläufen ein erheblicher Unterschied. Grundgesetze des Fernsehens wurden bewußt gebrochen, um die enorme Flexibilität, die durch den Einsatz der Computer entsteht, zu verwirklichen.
Den Kontakt zur Außenwelt erhält das Mediamobil über zwei Modems. Wann immer im Laufe des Einsatzes ein Problem auftaucht, kann über das öffentliche Fernsprechnetz Direktkontakt mit der Programmierzentrale Innsbruck aufgenommen werden, um Programme zu ändern oder neue Programme zu überspielen.
Konventionelle Technik ist im Mediamobil von Commodore zur Klimatisierung der Geräte eingesetzt. Zwei Klimaanlagen sorgen für Temperaturableitung. Wärme liefert die Dieselheizung sowie eine spezielle Fußbodenheizung.
(o. h.)

Jusos im Netz

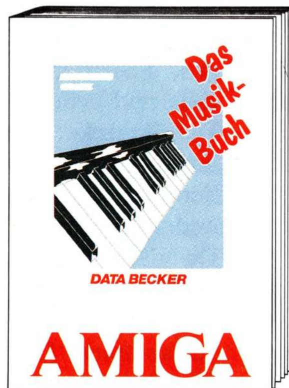
Jungsozialisten tun sich schwer, Technikfeindlichkeit mit politischer Progressivität unter einen Hut zu bringen. Neuerlichen Beweis dafür lieferte das Forum Jugend und Technik zum Thema Computerisierung der Politszene. Das Ergebnis tendierte gegen Null.

Können Computer rot werden?" fragten sich Jusos und Juso-Unabhängige Mitte April in der Bonner SPD-Zentrale. „Dr." Wau Holland, selbsternannter Alterspräsident des einschlägig bekannten Chaos Computer Clubs (CCC) und schillerndes Idol der Computerfreaks forderte eine linke Mailbox, um politisch relevante Informationen und Meinungen in der Juso-Szene auszutauschen. Letztendlich strebt er jedoch offene Netze mit dynamischen Datenbanken an, aus denen jeder Bürger sämtliches verfügbares Wissen lesen und auch selbst Informationen eingeben kann. Wau begründet diese Strategie mit der Feststellung, Computer seien Informationswaffen und Wissen sei schließlich Macht. Darauf ließ sich die Phalanx der Gegner nicht ein. Die Kontras ziehen es vor, auf herkömmliche Art und Weise zu kommunizieren und dem Gegenüber in's Auge zu blicken. Gleichzeitig wächst die Befürchtung, Beschäftigung mit Computern instrumentalisiere das gesamte Denken. Unkenntnis erzeugt Angst, Angst erzeugt Ablehnung. Und wo Wissen fehlt, gedeihen Vorurteile und Klischeevorstellungen. Technikorientierte Jusos deuten die Zeichen der Zeit dahingehend, daß sich Computer und elektronische Kommunikation nicht wegdiskutieren lassen. Aus dieser Einsicht heraus erkannten EDV-Insider das politische Potential der Computerfreaks und der professionellen Anwender. Folglich erscheint es ihnen opportun, Computer fest in die politische Arbeit zu integrieren. Unschlüssigkeit herrschte über das didaktisch wertvollste Verfahren. Ganz aus dem Ruder geriet die Diskussion zur These „spielerisch lernt es sich am besten". Die Runde ward gespalten. Während die Fortschrittsgläubigen Computerspiele als geeignete, zwanglose Annäherung an das elektronische Monster empfanden, lehnten radikale Technophobe dies als kommunikationstötend und realitätsentfremdend ab. Arno Hesse, Leiter der Projektgruppe Jugend und Technik hatte seine Not, die Genossen beim Thema zu halten. Wie verhext entartete fast jede fachliche Auseinandersetzung zur politischen Grundsatzdiskussion. Kein Wunder, daß im Plenum keine Klärung und damit kein einheitlicher Entschluß zustande kam.
(Ute Bahn)

RUND UM DEN

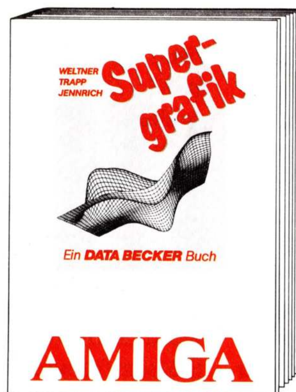


Licht und Schatten.
Dieses Buch öffnet Ihnen den Weg in eine faszinierende Welt. Anhand ausgefeilter Grafikalgorithmen erzeugen Sie phantastische Bilder: naturgetreue Spiegelungen, Licht und Schatten. Alles absolut realistisch und automatisch berechnet. In allen Auflösungen mit bis zu 4096 Farben!
Amiga 3-D-Grafikprogrammierung
Hardcover, 283 Seiten
inkl. Diskette, DM 59,–

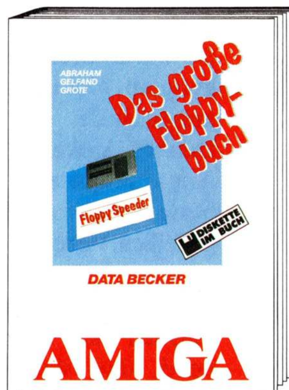


Da steckt Musik drin.
Zaubern Sie zarte Klänge oder heiße Rhythmen aus Ihrem Amiga – mit dem Musikbuch. Hier werden Sie zu einem Komponisten ausgebildet, der nicht nur die notwendigen Grundbegriffe der Musiktheorie beherrscht, sondern auch modernste Technik einzusetzen weiß: Musikprogramme wie Sonix, DeLuxe Construction Set oder Audio Master, Sampler, MIDI-Interface und, und, und.
Amiga-Musikbuch
Hardcover, ca. 300 Seiten,
DM 49,–
erscheint ca. 6/88

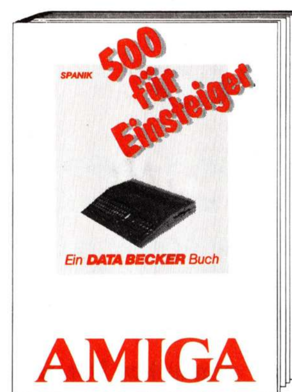




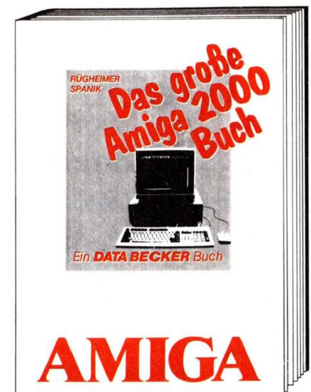
Das Buch zum Thema Nr. 1.
Grafik auf dem Amiga – mit dem entsprechenden Know-how ist hier fast alles möglich. Dieses Buch bietet es Ihnen: Nutzung der Libraries, die Register der Grafik-Chips, Aufbau und Programmierung von Screens, Windows, Halfbrite und Interlace aus BASIC und C uvm.
Amiga Supergrafik
Hardcover, 686 Seiten
inkl. Diskette, DM 59,–



Alles zur Amiga-Floppy.
Wie umfassend die Informationen eines Floppy-Buches von DATA BECKER sind, dürfte ja wohl bekannt sein. Im Amiga Floppybuch finden Sie darüber hinaus noch eine ganze Menge mehr, z. B. ein Superkopierprogramm, einen Floppyspeeder, einen Diskmonitor...
Amiga Floppybuch
Hardcover, 398 Seiten
inkl. Diskette, DM 59,–



Gleich loslegen.
Bei „500 für Einsteiger" heißt es aufstellen, anschließen und sofort loslegen. Schnell und für jedermann verständlich zeigt Ihnen dieses Buch, was Sie mit Ihrem neuen Rechner so alles anstellen können. Workbench, AmigaBASIC, CLI und DOS – schon bald wissen Sie, worauf es ankommt.
Amiga 500 für Einsteiger
343 Seiten, DM 39,–

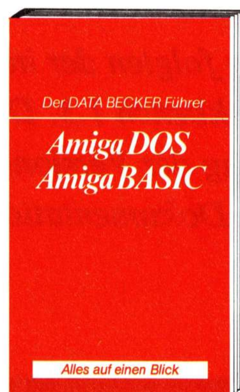


Ihr ständiger Begleiter.
Haben Sie einmal mit dem großen Amiga-2000-Buch gearbeitet, wird es sicherlich seinen festen Platz neben Ihrem Amiga behalten. Denn dieses Buch bietet Ihnen mehr als eine detaillierte Einführung. Vom Laufwerkseinbau bis hin zum Kickstart im RAM wird hier echtes Profi-Wissen vermittelt.
Das große
Amiga-2000-Buch
Hardcover, 684 Seiten, DM 59,–

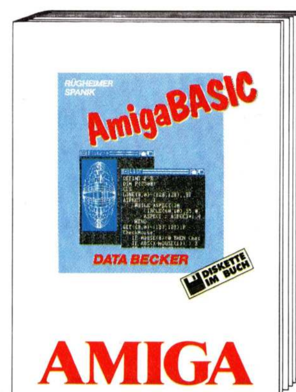
DATA BECKER
Merowingerstr. 30 · 4000 Düsseldorf · Tel. (02 11) 31 00 10

AMIGA!

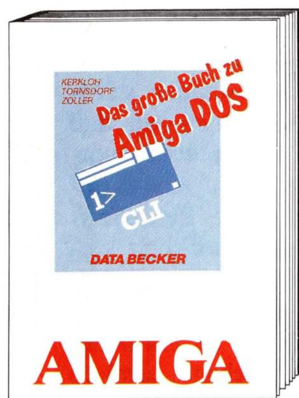




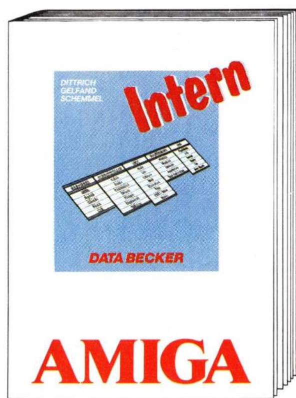
Alles auf einen Blick.
Der DATA BECKER Führer zu AmigaDOS und AmigaBASIC – das schnelle Nachschlagewerk für jeden Amiga-Anwender. Alle Befehle und Kommandos finden Sie hier auf einen Blick.
Der DATA BECKER Führer zu AmigaDOS & -BASIC
269 Seiten, DM 24,80



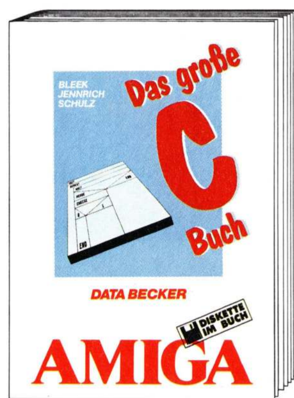
AmigaBASIC komplett.
Dieses Buch zeigt auf über 770 Seiten, worauf es beim Programmieren in AmigaBASIC ankommt. Natürlich mit jeder Menge interessanter Programmbeispiele, die auch gleich auf Diskette mitgeliefert werden. Ebenfalls im Buch: eine detaillierte Beschreibung des AC-BASIC-Compilers.
AmigaBASIC
Hardcover, 775 Seiten
inkl. Diskette, DM 59,–



Runter von der Workbench.
Rein ins AmigaDOS. Denn hier eröffnen sich ganz neue Möglichkeiten: Umlenken von Ein- und Ausgabe, mit RAM-Disk und CLI arbeiten, eigene CLI-Befehle programmieren, Batch-Dateien, Multitasking mit dem CLI, STARTUP-Sequenz... Das Know-how hierzu finden Sie im großen Buch zum AmigaDOS.
Das große Buch zu AmigaDOS
Hardcover, 310 Seiten, DM 49,–



Know-how aus erster Hand.
Amiga Intern – der ganze Rechner komplett in einem Buch: 68000-Prozessor, CIA, Blitter, Customchips, die Strukturen von EXEC, I/O-Handhabung, Verwaltung der Ressources, EXEC-Base, resetfeste Programme, IFF-Format, Programmierung der EXEC- und DOS-Routinen... Eben ein typisches Intern von DATA BECKER.
Amiga Intern
Hardcover, 639 Seiten, DM 69,–



Profi-Programme in C.
Wenn Sie an C Spaß gefunden haben, gibt Ihnen das große C-Buch den letzten Schliff. Denn hier erfahren Sie nicht nur, wie ein C-Compiler arbeitet und wie Sie selbst die schwierigsten Probleme in C lösen, sondern auch, wie Sie eine optimale Benutzeroberfläche entwickeln.
Das große C-Buch zum Amiga
Hardcover, 682 Seiten
inkl. Diskette, DM 69,–

Coupon!

HIERMIT BESTELLE ICH

NAME, VORNAME

STRASSE

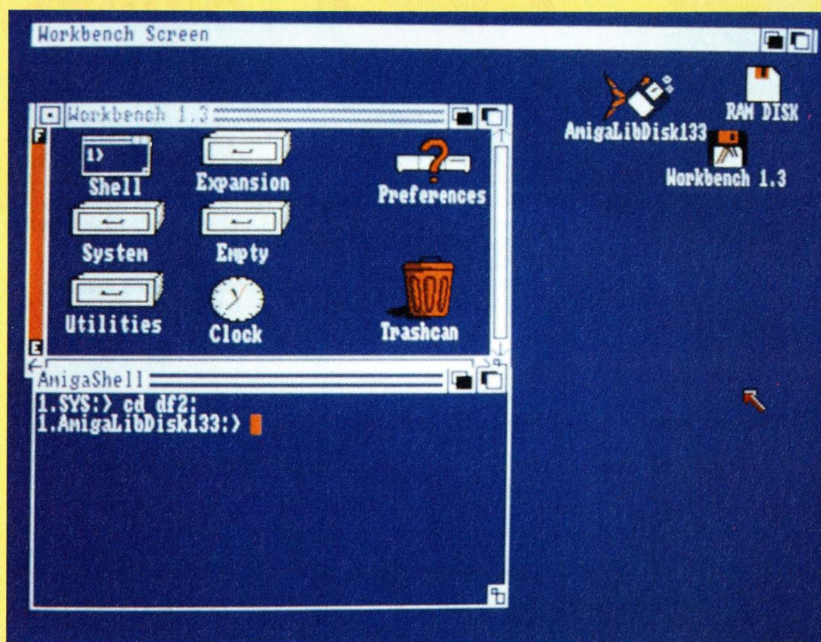
ORT

MEIN COMPUTER

zzgl. DM 5,– Versandkosten unabhängig von der bestellten Stückzahl
☐ per Nachnahme ☐ Verrechnungsscheck liegt bei

Ende der Spekulationen über Kickstart und Workbench 1.3

Gerüchte folgten der inoffiziellen Einführung von Kickstart und Workbench 1.3. Denn leider hat Commodore in froher Erwartung der Versionen 1.4 nicht nur die Software, sondern auch die Dokumentationen zurückgehalten.



CLI meldet sich jetzt als komfortable AmigaShell.

Von „die Diskette nicht wert" bis „unbezahlbar" reichen die Beurteilungen. Kaum einer scheint sich die Mühe gemacht zu haben, einen genaueren Blick auf die entsprechenden Disketten zu werfen. Ganz abgesehen davon, muß es sich im Normalfall ohnehin um Raubkopien gehandelt haben. Die allgemein verbreitete Version Gamma 4 wurde nur an registrierte Entwickler ausgegeben und eine Weitergabe strikt untersagt.

Zur Verteidigung von Commodore muß zu diesem Thema bemerkt werden, daß es lediglich Verwirrung stiftet, wenn Otto Normalverbraucher in den Besitz einer noch nicht fertiggestellten Version des Betriebssystems gerät. Klingelnde Telefone und unzählige Stunden verschwendeter Zeit, die an anderen Projekten fehlen, sind die Folge, wenn durch veraltete Versionen Fehler und Datenverluste entstehen. Anstatt sich seiner eigenen Schuld bewußt zu sein, wird wieder einmal Commodore angeklagt, unzuverlässige Systemsoftware zu produzieren. Commodore-Mitarbeiter in Frankfurt und Braunschweig können sicherlich ein Lied davon singen.

Wenig Spaß mit raubkopierten Gammaversionen

Soviel der Vorrede, aber viele Amiga-Anwender scheinen der Ansicht zu sein, daß ihnen böswillig etwas vorenthalten wird. Aber: Sobald eine neue Betriebssystem-Version wirklich fertiggestellt ist, wird sie auch offiziell erhältlich sein. Auch 1.2 war in Form des Enhancer-Packages legal zu beziehen, das vorherige Chaos war absolut unnötig. Die neue Version 1.3 besteht aus Kickstart (der ROM-Software), Workbench und Extras. Dabei kann Workbench 1.3 mit Kickstart 1.2 betrieben werden, und umgekehrt läuft die alte Workbench 1.2 auch mit Kickstart 1.3. Es ist also nicht zwingend notwendig, beim Amiga 500 und Amiga 2000 die ROMs auszutauschen, um in den Genuß eines verbesserten Systems zu kommen.

Die Unterschiede zwischen Kickstart 1.2 und 1.3 bestehen darin, daß der Anwender nun die Möglichkeit zum sogenannten „Autoboot" hat, also von einem anderen Laufwerk als DF0: das System hochfahren kann. Das Gerät muß natürlich dafür vorbereitet sein. Trotzdem ist es weiterhin möglich, von DF0: zu booten, da eine mit korrektem Bootblock versehene Diskette im internen Laufwerk sich immer über etwaige Autoboot-Devices hinwegsetzt. Funktionieren wird dieses Feature für Festplatten allerdings nur mit DHx:-Devices, also Harddisks auf Amiga-Seite, und auch nur, wenn der noch nicht erhältliche neue Controller vorhanden ist. Das Vorziehen einer in DF0: steckenden Bootdiskette hat einen guten Grund. Andernfalls könnten Sie bei einem Lesefehler auf Festplatte erst einmal Ihren Rechner demontieren . . .

Freie Wahl zwischen Festplatten- und Laufwerk-Boot

Um die neuen Routinen für das Autoboot noch ins Betriebssystem zu quetschen (es ist dort schon recht eng), wurden all die versteckten Texte entfernt. Wirklich alle? „Zumindest die, von denen wir wissen", meinte Andy Finkel auf eine entsprechende Frage.

Ansonsten ist Kickstart 1.3 ABSOLUT UNVERÄNDERT! Es wurden keine Bugs behoben und auch sonst nichts geändert. Wer also keinen Autoboot braucht, muß sich um neue ROMs keine Gedanken machen, ihm entgeht wirklich nichts!

Mit der Workbench auch die Extras 1.3

Dies ist die System-Software, die ohnehin auf Diskette ausgeliefert wird und daher jedem Anwender zur Verfügung steht.

In aller Munde ist die neue Druckersoftware von 1.3, nicht zuletzt deswegen, weil von unabhängiger Seite ebenfalls schnellere Druckertreiber entwickelt wurden und sich nun natürlich die Frage stellt, für welche Lösung man sich letztendlich entscheidet.

Während unter Workbench 1.2 das „printer.device" bei der Grafikausgabe in erster Linie damit beschäftigt war, dem spezifischen Treiber jeden Punkt in jeder Farbe einzeln zu übergeben, und so der Druckkopf eines hochauflösenden Druckers alle paar Minuten einmal in Aktion treten konnte (von Druckern wie dem HP-Laserjet oder dem Thermotransferierer von CalComp wollen wir hier gar nicht erst reden), so wurde durch einige konzeptionelle Änderungen die Software jetzt so weit verbessert, daß die Druckzeit fast nur noch vom Drucker und nicht mehr vom Rechner abhängt — und dies, obwohl die spezifischen Treiber noch immer in C geschrieben sind. Nach Aussagen der Amiga-Entwickler kann die Geschwindigkeit für schnelle Drucker noch verdoppelt werden, wenn gewisse Segmente in Assembler codiert werden.

Diese bestenfalls über 20-mal so schnellen Druckertreiber (im Vergleich zu 1.2) stellen den neuen Standard dar! Sie können ebenfalls wie die kopiergeschützten Notbehelfe (nur für Disk-Jockeys) HAM-Ausschnitte drucken und besitzen zusätzlich neue und verbesserte Algorithmen für die Ausgabe: Antialiasing für linienorientierte Grafik speziell von Textprogrammen (um den Treppeneffekt von Diagonalen zu reduzieren) und verschiedene Methoden zur Errechnung der Farbwiedergabe sowie Ganzzahlskalierung und weiteres mehr. Zu all diesen von Preferences aus wählbaren Optionen kommen noch verschiedene Druckdichten, die in sieben verschiedenen Stufen ausgewählt werden können.

Harte Disks und andere Härten

Vom Autoboot war ja schon die Rede: Einschalten und loslegen, ohne eine Diskette in DF0: zu haben. Doch auch ohne Festplatte kann man Gebrauch vom Autoboot machen — mit der neuen resetfesten RAM-Disk von Commodore („ramdrive.device")! Diese ist, ähnlich VD0:, als Zusatz zur konventionellen RAM-Disk gedacht und kann unter Kickstart 1.3 sogar zum Booten genutzt werden. Das heißt, daß Sie den Rechner durch CTRL-Amiga-Amiga zurücksetzen können und danach, ohne einen einzigen Diskettenzugriff, mit der Abarbeitung der Startup-Sequence begonnen wird. Das erinnert schon fast an die Zeiten, da Betriebssysteme noch vollständig im ROM waren, und läßt speziell beim absturzgeplagten Software-Entwickler durch schnelle Betriebsbereitschaft wahre Freude aufkommen. Sicherlich eine der genialsten Innovationen der neuen Workbench!

Doch das war beim Amiga ja nicht das einzige Diskettenproblem: Auch die Geschwindigkeit des Diskettenzugriffs wurde berechtigterweise bemängelt, da das AmigaDOS in erster Linie mit der schwerwiegenden Entscheidung beschäftigt war, was denn als nächstes wohl zu tun sei, anstelle sich endlich um die Übertragung der Daten zu kümmern. Durch das von Steve Beats entwickelte neue Fast Filing System (FFS) wurde ein Riesenschritt in die Richtung professioneller Anwendung getan: Durch geringfügige Änderungen im Blocklayout (aus denen nebenbei eine um 4.9 Prozent höhere Diskettenkapazität resultiert) und vollständiges Neuschreiben des Handlers in Assembler (die alte Version war BCPL) werden nun mit entsprechender Festplatte Übertragungsraten von weit über 600 Kilobytes pro Sekunde erreicht. Dies gelingt dadurch, daß Schreib- und Lesezugriffe — sofern möglich — in großen zusammenhängenden Stücken, und nicht blockweise, durchgeführt werden. Neues Handling des internen Cache beschleuni- ▶

gen auch den entnervenden Directory-Zugriff um das 5- bis 20-fache.
Eine Einschränkung ergibt sich im Betrieb des Fast Filing System mit dem Amiga-Controller A2090. Dieser installiert automatisch das alte Filing System für die erste Partition, so daß FFS nur für weitere Partitionen verwendet werden kann. Dafür besteht unter FFS nicht mehr das Limit von gut 51 MB pro Partition.
Auch ist es nicht möglich, von einem FFS-Device Autoboot durchzuführen, da das Disketten-Betriebssystem schließlich beim Reset-Vorgang schon zur Verfügung stehen muß. Schließlich muß das FFS erst von Diskette nachgeladen werden. Erst Kickstart 1.4 verspricht Abhilfe. Auch sollte zur Zeit das FFS nicht mit wechselbaren Disketten, etwa DF0: bis DF3:, kombiniert werden.

Neue Gerätetreiber, Libraries und ähnliches

Zuerst einmal wurde mancher Fehler in der RAM-Disk beseitigt. So stürzt der Rechner nicht mehr ab, wenn ein Zugriff über den Volume- (RAM Disk:) anstelle des Device-Namens (RAM:) erfolgt. Auch ein RELABEL wird nun anstandslos akzeptiert.
Für die Fans der Mathematik und Computergrafik sind die Routinen für doppelte Genauigkeit nun mindestens rund doppelt so schnell wie unter 1.2. Eine zusätzliche Library für transzendente Funktionen mit doppelter Genauigkeit ist neu hinzugekommen und entspricht funktionell der schon zuvor vorhandenen „mathtrans.library" für das Fast-Floating-Point-Format. Auch wird von diesen beiden Bibliotheken automatisch erkannt, wenn ein Fließkommaprozessor ins System eingebunden ist, so daß ein und dieselbe Version eines Programms selbsttätig auf einem Rechner mit 68881 schneller läuft.
Durch eine Erweiterung der „diskfont.library" wird es beim Arbeiten mit Diskettenfonts nun nicht mehr vorkommen, daß ein gelöschter Font trotzdem als vorhanden gelistet wird. Ein Programm zur Korrektur derart defekter Font-Verzeichnisse (FixFonts) ist ebenfalls vorhanden.
Der „Speak-Handler" ist ein DOS-Device ähnlich PRT, auf dem nichts gespeichert werden kann. Dafür liest Ihnen der Amiga beispielsweise durch ...
TYPE DEVS:MountList TO SPEAKER:
... den Inhalt der MountList vor!
Ein neuer Konsolen-Handler (NEWCON:) erlaubt nun, ohne daß sich das Anwendungsprogramm darum kümmern muß, solche Dinge wie Zeileneditierung und Wandern in der History, in Zeilen also, die zuvor eingegeben wurden.
Dazu kommen noch ein AUX- und ein PIPE-Handler, die Besitzer des MCC-Toolkit bereits bekannt sein werden. AUX ist ein Konsolen-Handler für serielle Übertragung, und PIPE legt temporäre Puffer, ähnlich (aber wirklich nur entfernt) Dateien in der RAM-Disk, für die Kommunikation zwischen Tasks an.

Mehr Möglichkeiten aus dem CLI

Aus den neuen Handlern AUX: und PIPE: kann man sicherlich im CLI den meisten Nutzen ziehen, so etwa für den Datenaustausch zwischen verschiedenen Kommandos und das Arbeiten in CLIs an seriellen Terminals. Auch eignet sich der NEWCON-Handler bestens für einen History-Modus, der das Wiederholen bereits zuvor eingegebener Kommandos ermöglicht.
Doch sind dies beileibe nicht alle Änderungen: Das CLI wird nun endlich durch eine neue Shell ab- und so der Benutzer erlöst. Diese Shell unterstützt residente Kommandos, Befehle also, die, einmal installiert, nicht bei jeder Eingabe erneut von Diskette geladen werden müssen und so bei jedem Aufruf weiteren Platz verschlingen (ähnlich Befehlen in der RAM-Disk).
Auch können Scriptfiles ohne EXECUTE ausgeführt werden, was zusammen mit kleinen Verbesserungen der Steueranweisungen die Verwendung von Batchdateien sicherlich wesentlich populärer machen wird. So kollidiert ein verschachteltes EXECUTE (oder eines mit Parametern!) nun nicht mehr mit einer schreibgeschützten Diskette.
Beim Start einer Shell werden bestimmte vom Anwender festgelegte Initialisierungen automatisch vorgenommen. In einem Scriptfile namens CLI- bzw. Shell-Startup lassen sich etwa die neuerdings vom Shell unterstützten ALIASES festlegen, auch das CLI-Prompt, nun wahlweise eine Kombination aus aktuellem Verzeichnis und CLI-Nummer, ist beliebtes Objekt solcher Startvorbereitungen.
Weiterhin stehen nun auch sogenannte „Environment-Variablen" zur Verfügung, Benutzer von Aztec-C oder ARP kennen diese ja bereits.

COPY kopiert jetzt wahlweise auch Kommentare, Datum und Uhrzeit sowie Protection-Flags. Die Angabe einer Puffergröße ist ebenfalls möglich.

Apropos Protection-Flags: Hiervon gibt es mittlerweile ganze acht Stück, so zusätzlich zur Kennung von Scriptfiles, reentrant Code und versteckten Dateien, wobei letzteres aber nicht vom CLI beachtet wird. Dafür beachtet das Fast Filing System aber mittlerweile endlich solche Flags wie „R" und „W". Diese Flags können durch den verbesserten PROTECT-Befehl nun auch einzeln verändert werden.

CHANGETASKPRI kann die Priorität anderer CLIs ändern.

Alle Flags, auch das schon seit 1.2 vorhandene Archive-Bit, werden nun von LIST angezeigt. Weitere Verbesserungen und Erweiterungen von LIST betreffen die Patterns: Sie werden nun auch ohne das berüchtige vorangestellte „P" erkannt, und das Ausgabeformat kann nun vom Anwender festgelegt werden.

Stellen Sie sich vor, daß Sie zwei Sätze an Dateien, etwa INCLUDE-Files, auf Unterschiede untersuchen wollen. Unglücklicherweise versteht Ihr DIFF-Befehl keine Patterns, so daß Sie nicht einfach eingeben können ...

DIFF old/? new/?

Mit dem neuen LIST kein Problem mehr. Geben Sie hierzu einfach an ...

```
LIST old/? TO Diff-
Script QUICK NOHEAD
    LFORMAT „DIFF old/
%S new/%S"
    EXECUTE DiffScript
```

Genial, nicht wahr? Weitere Verbesserungen lassen Sie nun auch die Blockgröße einer Datei ausgeben, und eine Selektion zwischen Dateien und Verzeichnissen ist ebenfalls möglich.

Vorsichtige Anwender können per Software-Verriegelung (LOCK) den Schreibzugriff auf Ihrer Harddisk setzen und so versehentliches Löschen unterbinden.

ASSIGN kennt eine neue Option UNMOUNT, um Devices wieder entfernen zu können, SEARCH kann auch die Existenz und den Pfad einer Datei „irgendwo auf der Diskette" feststellen.

Wen die Geschwindigkeit von FORMAT bisher gestört hat, dem kann eine frohe Botschaft verkündet werden: Es ist nun möglich, eine bereits formatierte Diskette ähnlich dem „Formatieren ohne ID" der CBM- und 64er-Floppies zu bearbeiten.

Nostalgisches Formatieren ohne ID

Es werden lediglich Root und Bitmap instandgesetzt, und schon kann die Diskette (oder auch Harddisk) neuen Zwecken zugeführt werden.

Auch wurden viele Feinheiten korrigiert. So stehen anstelle unzähliger einbuchstabiger Optionen (etwa für DIR oder TYPE) nun auch Schlüsselwörter zur Auswahl (TYPE HEX anstelle von TYPE OPT H zum Beispiel), die sich leichter merken lassen. INFO läßt sich auf bestimmte Devices anwenden und MOUNT kann nun auch auf fremde MountLists zugreifen. VERSION bestimmt die genaue Releasenummer eines separaten Betriebssystemsegments, etwa einer Library oder eines Device, ECHO macht nicht zwangsweise in der nächsten Zeile weiter und der INSTALL-Befehl überprüft Disketten auch auf Viren.

Preferences speichert das aktuelle Datum samt Uhrzeit automatisch in eine eventuell vorhandene batteriegepufferte Uhr des A500 und A2000.

„SetAlert" korrigiert einen Betriebssystemfehler in Amigas mit $C00000-RAM, also speziell dem A500 und A2000. Hier kam es bei einem „Recoverable Alert" fälschlicherweise zu einem Neustart des Systems, was nun nicht mehr der Fall ist.

Und die FastFonts von Charlie Heath zur Beschleunigung der Textausgabe sind gleich als Gratisbeigabe zu finden.

Nachsitzen für den Programmierer

Jetzt heißt es wieder: Lernen! Die spezifischen Druckertreiber müssen völlig neu entwickelt werden, um die neuen Features wirklich alle nutzen zu können. Es empfiehlt sich übrigens nicht mehr, die Include-Files zu 1.2.1 (wie etwa bei Lattice-C-4.00 mitgeliefert) zu verwenden, da diese nicht mehr den aktuellen Umständen entsprechen und zum Teil Daten enthalten, die erst unter Kickstart 1.4 aktuell werden. Sollten Sie keinen Zugang zu 1.3-Versionen haben, so verwenden Sie lieber die alten 1.2-Includes.

Als kleines Bonbon wird dafür auch eine neue Debugging-Library für den Parallelport mitgeliefert.

Kickstart und Workbench 1.3 stellen natürlich nicht das Ende der Evolution des Amiga-Betriebssystems dar.

Ausblicke auf 1.4

So wird es unter 1.4 zu massiven Änderungen der ROM-Software kommen: Fehler werden (endlich) behoben werden und auch Unterstützung für die neue Grafikhardware, den Hedley-Monitor und die neuen Customchips, sind nötig. Das Fast Filing System kommt ins ROM und steht dann auch für Autoboot und 3.5-Zoll-Disketten zur Verfügung. Die Workbench-Software wird dafür entfernt (das spart 32 Kilobytes) und zukünftig wohl auf der gleichnamigen Diskette zu finden sein. Dafür stehen auch hier einige Erweiterungen an, um dem Nur-Anwender mit Komfort entgegenzukommen und beispielsweise Workbench-Menüs zu definieren und Icons auch von Programmen zu nutzen, was bisher nur auf illegalen Umwegen machbar war. Sogar an Unterstützung für völlig unabhängig entwickelte Grafikhardware ist gedacht. „Das ist zwar hart, aber nicht unmöglich", lautete diesbezüglich der Kommentar von Andy Finkel. Unterstützung für Netzwerke, vermutlich mit Blick auf den für nächstes Jahr geplanten Amiga 3000, ist nach Informationen von Dale Luck ebenfalls in der Mache. Aber endgültig ist über die Zusammensetzung noch nicht entschieden, denn schließlich muß erst einmal 1.3 wirklich vollendet werden.

(Ralph Babel)

Nur für Insider!

Eure Idee, deutschen Computerspielern Baseball und die Spielregeln beizubringen, finde ich super. Aufmachung, Grafik und Erklärungen sind professionell, — über alle Kritik erhaben. Outsider finden durch den Artikel vielleicht Geschmack an Baseball und möchten Baseball spielen lernen. Das kann man aber nicht am Computer! Ich bin Amerikaner und lebe seit zehn Jahren in Deutschland. Von meinem 12. bis zum 24. Lebensjahr habe ich Baseball gespielt. Baseball selber spielen macht unheimlich viel Spaß. Aber ein Außenstehender kann kaum eine Beziehung zu dem Spiel entwickeln. Es ist ähnlich wie beim Schachspiel. Man muß es selber spielen. Zuschauen macht wenig Spaß, und man lernt dabei nichts. Ich habe selbst versucht, anderen die Baseballregeln zu erklären. Aber auch da konnte ich nur raten, es einfach selbst zu spielen. Deshalb glaube ich, daß deutsche Computerfreaks, die Baseball nie gespielt haben, nicht so viel Spaß haben. Für sie haben schließlich Ballparks wie „Fenway Park" gar keine Bedeutung. Für uns schon. Hier in Deutschland ist ja eher Fußball so bekannt und verbreitet wie Baseball in den USA.

Paul Hallgrimson,
Langen

Kein „ultimatives" Baseball!

Mit viel Interesse habe ich den Artikel über Earl Weaver Baseball gelesen. Aber, so sehr ich dieses Computerspiel schätze, so habe ich doch das Gefühl, es hätte besser sein können. Für „starters" (Baseballspieler, die das Spiel beginnen) existiert keine „infield fly"-Regel (sie besagt: wenn je ein Spieler auf der 1. und 2. Base steht, ist der Batter out, sofern er den Ball in die Luft geschlagen hat). Es fehlt auch die Regel, daß Ersatzpitcher mindestens zu einem Batter werfen müssen, bevor sie ausgewechselt werden können. Der „official scorer" (entspricht dem Schiedsspruch) ist in seiner Wertung sehr inkonsequent. Wenn Fehler (Ball fällt zwischen zwei Bases zu Boden) auftauchen, gibt er sie nicht an; er vergibt ein „double" (zwei Bases geschafft) für ein „single", und so fort. Die statistische Aufstellung hat bei mir noch nicht korrekt gearbeitet. Statistiken einer „six-game series" wiesen für ein Team mehr RBIs (=runs battered in; es kann maximal soviele RBIs geben wie runs erzielt wurden) als „runs" aus! Die Batting-Statistiken paßten nicht zu den Pitching-Statistiken (hits, walks und strike-outs). Ich hoffe, Electronic Arts wird meine Einwände bei einer Neufassung von Earl Weaver Baseball berücksichtigen. Ansonsten muß ein ultimatives Computer-Baseballspiel noch entwickelt werden.

N. Longardner,
Gahanna (USA)

Objekt-Modul in C?

Seit einiger Zeit besitze ich das Spiel „Barbarian", das eine sehr eindrucksvolle Startroutine hat. Wenn man sich nun das Directory dieser Diskette ansieht, so muß man feststellen, daß sie keinerlei Files enthält und ohne das Zutun vom AmigaDOS (also die Abarbeitung der startup-sequence) startet. Das ganze Spiel besteht nur aus Objekt-Modulen. Kann man so etwas auch in der Sprache C programmieren, und wenn ja, wie? Ich vermute eine Verbindung zum Bootblock dieser Diskette.

Und: Kann ich IFF-Bilder als Objekt-Modul in meine C-Programme einbinden?

D. Tappe,
Walluf

Spiele wie Barbarian greifen zu keiner Zeit auf Amiga-Routinen zu. Daher ist ein Direktstart möglich. Im eigenen Programm ist das zumeist nicht so. Wie IFF-Bilder eingebunden werden, wird in kommenden Folgen unseres Grafik-Kurses erläutert.

AT-Karte und Datentransfer

Als Besitzer eines Amiga 2000 möchte ich mir eine AT-Karte und ein zweites internes Laufwerk kaufen. Mit Interesse habe ich nun die Antwort von S. Rossow und R. Sobanski gelesen (AmigaWelt 2/88). Zu diesem Problem habe ich folgende Fragen:

1. Gibt es eine Softwarelösung, die es erlaubt, von der AT-Karte aus im internen 3½-Zoll-Floppy DF0 eine PC-formatierte Diskette zu lesen und zu beschreiben?

2. Gibt es eine interne 3½-Zoll-Zweitfloppy DF1, die einerseits mit der AT-Karte das neue AT-Format mit 1,44 MByte/Diskette lesen und schreiben kann, und das auch völlig kompatibel als DF1 im Amiga-Modus arbeitet?

Ist für dieses Problem evtl. eine Lösung mit Kickstart/WB 1.3 in Sicht?

Dr. M. Rudin, Meggen (Schweiz)

Zahlreiche Leserzuschriften zu diesem Thema geben Ihnen Hilfen und Lösungen.

Datentransferproblem lösbar

Betr.: 1/88, Leserbrief „Laufwerk für XT-Karte"

Es ist mit Verlaub ziemlicher Unsinn, beim Anschluß eines weiteren Laufwerks an die XT-Karte im A2000 mit MOUNTLIST und TRACKDISK.DEVICE herumzuexperimentieren. Man hat zwei Möglichkeiten: 1. Mitbenutzung eines vorhandenen Amiga-Laufwerks von MS-DOS aus. Dazu muß der Treiber JDISK.SYS auf MS-DOS-Seite in der Datei CONFIG.SYS eingebunden sein und auf Amiga-Seite das Programm PCDISK aus der PC-Schublade laufen, am besten startet man es per run-Befehl schon in der Startup-Sequence an. Das Reservieren von Datenbereichen für MS-DOS geschieht dann von MS-DOS-Seite mit dem JLINK-Befehl, der ausführlichst im Handbuch beschrieben ist.

2. Derselben Literatur kann man entnehmen, daß man auch hardwaremäßig etwas tun kann. An der Rückseite der XT-Karte findet sich ein passender Anschluß für eine externe Amiga-Floppy. Hier kann man also eine A1010 anschließen, die dann mit 720 KB unter MS-DOS läuft, voll kompatibel. (Bitte nur im ausgeschalteten Zustand umstecken!). In der CONFIG.SYS findet man schon eine entsprechende Zeile mit DRIVPARM. Wer unbedingt 5¼-Zoll-Format fahren will, kann auf eines der vielen Angebote von Zubehörfirmen (z.B. Kupke) zurückgreifen.

Dr. Peter Kittel,
Frankfurt

Problem PC-Ebene

1: Achim Ruez „Das FixHunk Problem" — 1/88

Das Problem befindet sich auf Fish Nr. 36 beziehungsweise Amicus Nr. 17. Die Funktionsweise ist relativ einfach: Amiga-Programmdateien bestehen aus drei verschiedenen „hunk"- (oder Codemodul-)Typen — code, data, bss; hunk_code kann im Fast-RAM-Bereich abgelegt werden; hunk_date muß in den meisten Fällen im Chip_RAM-Bereich liegen; hunk_bss werden als Arbeitsspeicher für die Ablage von zum Beispiel Lokalvariablen benutzt und können daher in der Regel auch im Fast-RAM liegen. Jeder Hunk-Typ enthält eine Kennung für den jeweiligen Typ und den „bevorzugten" Speicherbereich (Chip- beziehungsweise Fast-RAM). Ist die Kennung im Programm nicht vorhanden, wird ein Hunk automatisch in Fast-RAM geladen. In den meisten Fällen können Programme auf diese Weise für Benutzung mit Version 1.2 „gepatcht" werden.

2. Ralf Sobanski — 1/88

Sie schreiben, daß trackdisk.device im DEVS-Verzeichnis zu finden sei. Das ist nicht richtig! Trackdisk.device befindet sich im sogenannten Boot-ROM — sonst könnte keine Kickstart-Diskette (A1000) eingelesen werden. Bezog sich Ihre Anmerkung jedoch eigentlich auf die Daten „mountlist" — was unter Umständen aus dem weiteren Verlauf der Antwort möglich wäre, wenn man etwas umständlich denkt — dann stimmt die Angabe.

3. Ivor Nissen — 1/88

Sie schreiben, Sidecar sei nicht 1022 IBM-kompatibel; wenn Sie meinen, es gäbe einige, sehr wenige, Programme, die auf sidecar nicht laufen, mögen Sie recht haben. Da aber Sidecar das wegen seiner ausgezeichneten Kompatibilität berühmte PHOENIX-BIOS — genau wie die PCs auch — verwendet, ist diese Aussage äußerst irreführend und — angesichts der sonst in Zeitschriften bescheinigten 99,99-Prozent-Kompatibilität — falsch. Es sei denn, Sie verfügen über Informationen, die bisher nicht bekannt sind, wie zum Beispiel, welche Programme NICHT auf Sidecar laufen. Jene hätten Sie dann fairerweise nennen müssen.

4. Andreas Bast, „Undokumentierte Systemroutinen" — 2/88

Neben den von Ihnen erwähnten exec-Routinen, die vom (ROM)-Wack benutzt werden, gibt es eine Reihe solcher Routinen; fast alle sind anhand sogenannter „fd_files" ausfindig zu machen, denn sie werden in diesen Dateien als „private" gekennzeichnet. Die von Ihnen erwähnten Routinen befinden sich zwar im „exec.library", werden aber von „debug. library" benutzt (vergleiche Kickstart Guide S. 160 f). Dazu kommen — auch in „exec.library" — sämtliche Multitasking-Routinen, die systemintern benutzt werden. Jede zweite Routine in „CString.library" fällt ebenfalls in die Kategorie „undokumentiert" — sofern man nicht zum Beispiel das Handbuch zu Aztec C gelesen hat. Ein Vergleich mit dieser Quelle ergibt, daß es sich hier um Routi-

nen handelt, die vorher im „amiga-.lib“ zu finden waren. Außerdem sind die Routinen von „info-library“ und „ramlib.library“ nicht weiter dokumentiert — hierzu existieren auch keine „fd_files“; endgültiger Aufschluß ist hier nur mit einer Disassemblierung dieser Libraries möglich. Eine volle Dokumentation der zugänglichen Library-Routinen befindet sich auf der — mehrmals im Kickstart Guide erwähnten — „AutoDocs“-Diskette zu Version 1.2 (beziehungsweise vermutlich auch zu der angekündigten Version 1.2.1 a. k. a. 1.2). Leider wird diese Diskette offensichtlich nur an registrierte, kommerziell tätige Entwickler abgegeben; sie trägt — laut Auskunft von Dave Parkinson, dem Hauptverantwortlichen für Kickstart Guide — die Aufschrift „confidental — not for distribution“. Aus diesem Grund sah er sich — auf meine diesbezügliche Anfrage hin — nicht befugt, mir eine Kopie zukommen zu lassen.

PS: Ich verstehe nicht, daß es Probleme mit der Lieferung des Kickstart Guide geben soll. Ich habe mein Exemplar direkt bei Ariadne Software in London bestellt (Adresse in jeder britischen Amiga- beziehungsweise Commodore-Zeitschrift). Das Buch war binnen 10 Tagen da! Bei Zahlung mit Euroscheck sollte es keine Lieferprobleme geben.
D. Twigg- Flesner,
Gronau

Garantiefall?

Letztes Jahr im Juni legte ich mir einen Amiga 1000 zu. Ich bestellte Anfang dieses Jahres eine Golem-Ram-Box mit 2 MByte. Die Box verweigerte ihren Dienst. Also rief ich die Versandfirma an und reklamierte. Bereits eine Woche später bekam ich das neue Gerät, beiliegend die Versicherung, daß diese Box überprüft worden sei. Die Box funktionierte nicht. Ich rief wieder bei der Versandfirma an. Die Box sei in der Firma gelaufen, ich solle mal meinen Computer überprüfen lassen. Nun überprüfte ich meine Box auf einem anderen Amiga 1000. Da gab es nun wirklich keine Mängel an der Box festzustellen, der Amiga meldete sich einwandfrei mit 2,5 MByte. Am nächsten Tag brachte ich dann meinen Computer zu der Firma, wo ich ihn gekauft hatte. Nach einer Woche erhielt ich einen Kostenvoranschlag über DM 350.—, da der Expansionport defekt sei. Nun meine Fragen: Ist es möglich, daß der Expansionport, der von mir nie benutzt wurde, von selbst defekt wird, oder ist es wahrscheinlicher, daß er nie richtig funktioniert hat, was ich eher glaube? Da die Garantie seit einem Monat abgelaufen ist, möchte ich wissen, ob ich in diesem Fall dennoch einen Anspruch auf kostenlose Reparatur habe.
S. Pawlik, Duisburg

In diesem Fall haben Sie keinen Anspruch mehr auf eine kostenlose Reparatur, außer, Sie könnten beweisen, daß der Fehler von Anfang an bestanden hat. Generell dürfte es unwahrscheinlich sein, daß der Expansionport ohne jegliche Benutzung die Funktion einstellt. Auch mir erscheint die zweite Möglichkeit wahrscheinlicher, dennoch besteht ein Garantieanspruch nur 6 Monate nach dem Kauf. Sie sollten versuchen, sich mit der Firma auf dem Kulanzweg zu einigen.
(Salus)

FixHunk: Aus eins mach zwei

Betr.: 1/88, Leserbrief „FixHunk-Problem“
FixHunk ist ein Programm auf der Fish-Disk Nr. 36. Es ist tatsächlich dazu gedacht, sonst nur unter Kickstart 1.1 lauffähige Programme an 1.2 anzupassen. Dazu werden alle Datenbereiche (Hunks) zwangsweise in's Chip-Mem dirigiert, nur der Programmcode darf im Fast-Mem bleiben. Das hilft, weil bei 1.1 diese Unterscheidung beim Laden noch nicht gemacht werden konnte. Andererseits kann es im Chip-Mem jetzt etwas eng werden, oder die Unverträglichkeit kann noch an anderen Punkten gelegen haben, also wieder nur an Kopien ausprobieren!
Dr. Peter Kittel,
Frankfurt

Inkompatible Spiele

Time Bandit läuft nicht richtig. Und zwar tritt bei ihm dasselbe Problem auf, das schon bei Barbarian sehr störte: Auf dem Amiga 2000 kann die Tastatur nicht benutzt werden! Der einzige Unterschied zu Barbarian ist der, daß bei Barbarian die Tastatur überhaupt nicht reagierte und bei Time Bandit anscheinend nur das erste Zeichen übertragen und erkannt wurde. Ich kann das Spiel starten, anwählen, ob ich mit einem oder mit zwei Spielern spielen will und kann bei Erreichen der Highscore-Liste mit RETURN eine leere Zeile eintragen und dann wieder weiterspielen. Sobald ich jedoch mehrere Buchstaben eingeben möchte, sei es beim Highscore oder während des Spieles (dies ist ja unbedingt notwendig, da Elemente eines Adventures integriert sind), versagt die Tastatur ihren Dienst. Alle meine anderen Programme, arbeiten auf dem Amiga 2000 einwandfrei (von dem Problem des Fast-Memories mal abgesehen). Dies ist kein Fehler, der sich durch Abschalten des Fastmem beheben läßt. Als ich Barbarian kaufte, hatte ich an Psygnosis geschrieben, und erhielt als Antwort: Wenn ich einen 2000er besitze, müsse ich mit Maus spielen, die Tastatur ginge halt nicht. Meine Frage, die wohl auch andere Amiga-Besitzer interessieren dürfte: Wo ist der Unterschied im Konsolhandler zwischen Amiga 2000, 500 und 1000? (Auf diesen Rechnern laufen die Spiele nämlich einwandfrei). Kann man das irgendwie beheben? Ich habe in keiner Zeitschrift gelesen, daß es überhaupt einen Unterschied gibt. Ich bitte Sie, bei künftigen Tests nach Möglichkeit auf Inkompatibilitäten zwischen den einzelnen Rechnertypen hinzuweisen.
R.-P. Böhlke,
Frankfurt

Offene Systemarchitektur?

Offener Brief an Herrn Dieter Preiß, Leiter der Commodore-Entwicklungsabteilung in Braunschweig:
Zitat aus dem großen Amiga Jahrbuch 1987: „Amiga 2000, gebaut für die Zukunft . . .“ Zitat: „Das große Buch zum Amiga 2000“, Seite 288: „Der neue Baustil — offene Systemarchitektur“ . . . Offene Systemarchitektur sagt aus, daß ein Computer so konzipiert und gebaut ist, daß „Erweiterungen und Ergänzungen jederzeit leicht möglich sind“ . . . dieses Feature des Amiga 2000 wird von Commodore in der Werbung kräftig unterstrichen. In Fachmagazinen, wie „AMIGA“, „AMIGA SPEZIAL“ und in der „AMIGA WELT“ wurde der Amiga 2000 für seine Ausbaubarkeit — „Das System mit Zukunft“ — in höchsten Tönen gepriesen.
Genau aus diesem Grund haben wir uns im November 1987 für den Amiga 2000 entschieden. Da wir beide in der Werbung tätig sind, — als Creative und Art-Director —, und wir auch zu Hause in Profiqualität arbeiten wollen, waren die versprochenen, künftigen Erweiterungsmöglichkeiten im professionellen Bereich das wichtigste Kaufargument.
Und jetzt stehen wir da, mit unserem Amiga 2000. Von den Werbeversprechungen ist nichts übriggeblieben als ein bitterer Nachgeschmack: Schon zwei Monate nach unserem Kauf kam nämlich der Amiga 2000 B auf den Markt. Unser 2000er gehörte über Nacht zum „alten Eisen“! Wie wir den oben angeführten Magazinen entnehmen können, wird es demnächst einen neuen Grafikchip für eine Auflösung 640 × 512 Non Interlace geben. Zitat aus der AMIGA WELT, Ausgabe Mai/Juni 1988: Frage „Lassen sich die vorhandenen Amiga 2000 mit dem neuen Chip nachrüsten?“ — Antwort Preiß: „Ja, das gilt jedoch nur für die 2000er mit dem B-Board . . .“ Des weiteren gibt es ein neues Genlock zum Einbau in den 2000er. Zitat AMIGA WELT: „Leider arbeitet dieses Gerät nur mit dem B-Board“.
In diesem Sinne, — nur noch der Amiga 2000 B ist für die Zukunft gerüstet, — gehen die diversen Publikationen weiter. Als Besitzer des „alten“ 2000ers müssen wir uns mit den weniger professionellen Zusatzkarten begnügen. Was wir nun wirklich nicht einsehen! Wir fühlen uns von dieser wenig konsumentenfreundlichen Marketing-Strategie um unser gutes Recht betrogen! Noch dazu, weil wir dieses unbefriedigende Gerät nur mit großem Verlust loswerden können: Wer kauft schon zu einem akzeptablen Preis eine hohle Ente? Oder wissen Sie einen Dummen?
Mit ganz schön viel Wut im Bauch,
Dr. Ulrike und Leonard Swennen,
Wien

Gentechnologen am Werk

Offensichtlich gibt es Amiga-User, die alles daransetzen, einen eigenen Virus in die Computerwelt zu setzen. Wenn die eigenen Programmierfähigkeiten dazu nicht ausreichen, bedient man sich eines vorhandenen Virus und ändert mit Hilfe eines Disketten-Monitors den Text. „Another future of programming has begun on Amiga!!!! Don't worry about our great V I R U S!!! Spread by Micro-Master on CCW and Odie from AEK!!!.“ Läßt diese lautstarke Prahlerei auf einen neuartigen Virus schließen, entpuppt sich die Angelegenheit bei näherem Hinsehen als primitive Variante des SCA-Virus. Diese Änderung führt aller- ▶

dings dazu, daß eine solche „Mutation“ von den bisherigen Detektoren nicht erkannt wird. Bei dieser Sachlage kann nur geraten werden, zur Virussuche einen Diskettenmonitor zu verwenden. Mit dem Monitor sieht man sich nun die ersten Blöcke der Diskette, die sog. Boot-Blöcke, an. Block 0 entspricht den Parametern Cylinder = 0, Track = 0, Sector = 0, während Block 1 durch die Werte Cylinder = 0, Track = 0, Sector = 1 bestimmt ist. Ist die Diskette infiziert, so findet man in der Regel innerhalb der ASCII-Darstellung der Sektoreninhalte einen lesbaren Text, mit dem sich der Virus zu erkennen gibt. Auf diese Weise muß man nun sämtliche Disketten überprüfen. Bei infizierten Disketten muß der Bootsektor mit dem CLI-Befehl „Install <Laufwerk>“ überschrieben werden. Bei einigen Programmen wie z. B. „Barbarian“ oder „DeluxePaint II“ enthalten die Bootsektoren wichtige Programmdaten. Nach dem „Install“-Befehl sind diese Programme nicht mehr lauffähig. Sie können diese Programme jedoch auch mit Virus weiterverwenden, gewissermaßen unter Quarantänebedingungen.
Ralf Kratzer,
Minden

Spiel-Zauber

Ich lese Ihre Zeitschrift seit der ersten Ausgabe, und finde sie super, besonders Grafik, Business, Tips & Tricks, auch die Listings sind gut. Wie Sie sicher schon wissen, ist die IFF-Norm ein Pluspunkt der Amiga-Software. Und das nicht nur bei Grafik- und Soundsoftware, sondern auch bei Spielen. So lassen sich Grafiken als auch Sound bei einigen Spielen verändern (geniale Idee!!!). Ich habe hier einige Tips zu „Testdrive“. Wem die Abbildungen der Autos zu langweilig sind, oder wer einen eigenen Titel machen will, kann das entsprechende Feature in Deluxe Paint II laden und verändern. Das geht folgendermaßen: Sicherheitskopie von „Testdrive“ anlegen (wichtig!), DPaint laden (Lo-res, 32 Farben); „Testdrive“-Diskette einlegen, Funktion „load“ im Picture-Menue anwählen; nach Df0: wechseln in Laderequester „cars“ (Autos) oder „pics“ (Titel). Dann klickt man Dateien mit der Endung „.st“ (Autos) oder „.shp“ (Cockpits) an und lädt. Das Bild ist manchmal etwas verschoben, und man muß es mit den Cursor-Tasten zentrieren (Merken Sie sich, in welcher Richtung und wie oft die Tasten gedrückt wurden!). Jetzt erscheint das Bild. Sie können es nun beliebig verändern, aber nicht vergrößern. Farbfelder können mit „Palette“ oder „RestorePalette“ verändert, gestaucht, gebogen werden. Sie können auch das Cockpit wie beim „Knight Rider“ verändern. Abschließend mit den Cursor-Tasten an die richtige Stelle verschieben und unter gleichem Namen abspeichern.
Übrigens: Diese Tips können Sie in ähnlicher Weise auch bei „Emetic Skimmer“ und „Karate King II“ anwenden.
Und hier ein kleiner Spieletip zu „Shadowgate“: Werwölfe mögen kein Silber.
Andy Weisner,
Hilden

Mastermind's Schönheitsfehler

Betr.: RUN Sonderheft 3/87 AmigaWelt; Mastermind-Listing, Seite 131ff.
Hilfe, mein Computer mogelt! Wir besitzen erst seit kurzem einen Amiga 500 und sind daher alles andere als Könner. Ich gebe sehr gerne Listings ein, und wenn sie dann auch noch das tun, was ich will, freue ich mich sehr.
Das eigentliche Problem: Mastermind's Spielablauf funktioniert, aber mit den Farben kennt sich der Computer nicht aus. Wenn ich acht oder mehr Farben einstelle, kann ich zwar die Farben richtig erraten, aber der Computer erkennt sie nicht als richtig an. Was mache ich falsch?
Luzia Leitpet,
Wien

Nachdem ich 14mal die falsche Lösung ausgedruckt hatte, entdeckte ich den Fehler. Auf Seite 135, erste Zeile, unter dem Label „Spielende“ muß es heißen:

```
LINE (a,32)—
(a+32,48),Rate.Code(t)
,bf:a=a+56
```

Statt Rate.Code war gedruckt:
Loesung(t)
A. Turnaus,
Hof

CeBIT-News

Ihre Zeitschrift gefällt mir ausgesprochen gut. Sie ist recht übersichtlich gegliedert, was man von (einigen wenigen) anderen Zeitschriften nicht gerade behaupten kann. Vielen Dank auch für den ausführlichen Messebericht; jetzt weiß ich wenigstens, was ich auf der CeBIT alles übersehen habe!
C. Eisen, Wrestedt

Erfolgreiche Virenkur

Euren Virenchecker aus Heft 2/88 habe ich erfolgreich abgetippt (für das nächste Listing werde ich mir die Eingabehilfen checksum und NanoMon besorgen). Mit Erstaunen stellte ich fest, daß sich auf meiner Diskette ein Programm befand, das sich im Vergleich zu der Schwemme von Virenärzten (meist gegen gutes Geld) als echte Perle erwies. Ganz offensichtlich werden nicht nur der Bootblock, sondern auch interne Register überprüft. ViCheck erwies sich so auch gewappnet gegen Mutanten neueren Datums, und ich hoffe auch gegen zukünftige Bootblockzombies. Besonders toll empfand ich den Komfort: Separate Diskettenüberprüfung erübrigt sich, da das Programm, einmal in die Startup eingebunden, jeden Eindringling meldet. Erstaunlich, was in meiner Diskettenbox für Teufelchen schlummerten! Vielleicht erlahmt angesichts solch eines hervorragenden Tools das Interesse gewisser Programmierer, die User per Bootblock das ganze Jahr über in den April zu schicken. Ihr habt meiner Meinung nach mit dem im Heft etwas verschämt plazierten Programm ganz schön Underdog betrieben.
Klaus Schulze,
Düsseldorf

Zum Teil ohne Kicks

Betr.: Leserdiskette AmigaWelt 2/88
Auf meine Bestellung der Leserdiskette zum Heft Nr. 2/88 wurde mir heute das entsprechende Päckchen zugestellt. Nach sofortigem Erstellen einer Sicherheitskopie, nahm ich die Disk gleich unter die Lupe. Nach Beendigung meiner Expedition mußte ich jedoch eine furchtbare Feststellung machen. Alle meine kleinen Helfer „Beep“, „Screenoff“, „Hello“, „Settime“ und „Time“ blieben für mich aus unerklärlichen Gründen verschollen.
D. Habinger, Pforzheim

Wegen einer inhaltlichen Änderung im Heft 2/88 in letzter Minute, sind auf einem Teil der Leserdisketten andere Amigakicks veröffentlicht als im Heft. Davon ist lediglich der erste Teil der Versendungen betroffen. Auf der Leserdiskette zu Heft 3/88 finden Sie zusätzlich die kompletten Amigakicks 2/88. Da es sich um Billigst-Software handelt, ist der Service-Firma ein Umtausch nicht möglich. In Einzelfällen hilft die Redaktion.

COBOL-Compiler für Amiga 1000?

Ich studiere an der Fachhochschule Betriebswirtschaft mit der Fachrichtung Datenverarbeitung. Die bevorzugte Programmiersprache ist hierbei COBOL. Um meine Programme auch zuhause austesten zu können, benötige ich einen COBOL-Compiler für den Amiga 1000. Bislang konnte ich jedoch keinen finden. Vielleicht können Sie mir weiterhelfen; möglicherweise existiert auch ein Public Domain-Compiler.
Josef Weber,
Hösacker

COBOL ist eine Programmiersprache für „historische“ Computer und nutzt die vielfältigen Talente des Amigas nicht aus. Uns ist nicht bekannt, ob es einen COBOL-Compiler für Amigas gibt. Wer einen Amiga-COBOL-Compiler kennt, möge sich an uns wenden.

Laufwerk-Killer?

Ist es eigentlich möglich, daß durch die Kopierschutz-Abfragen bei den neuen Programmen das Laufwerk beschädigt wird? Ich habe nämlich bei einigen Programmen — nagelneu gekauft — festgestellt, daß diese sehr rabiat mit meinem Laufwerk umgehen.
Da stellt sich mir auch die Frage, wenn wirklich mal ein Laufwerk kaputt geht, kann man die Softwarehersteller für den Defekt haftbar machen. Es reicht doch voll aus, wenn ein Wort aus der Anleitung abgefragt wird. Mir ist auch schon mal ein Programm nach Abspeichern des Highscores auf der Originaldiskette kaputtgegangen. Habe ich da Anspruch auf Ersatz?
P. Siess,
München
Haben andere Leser ähnliche Erfahrungen gemacht? Bitte schreiben Sie uns.

Im Interesse aller Leser:
Wer Antworten auf veröffentlichte Fragen weiß, möge sein Wissen allen zugänglich machen. Wir möchten die Rubrik Leserbriefe weniger als Lobes- und Meckerecke verstanden wissen, sondern als Forum für Probleme und Wissensaustausch.
Die Redaktion

Gaststar bei Miami Vice

Bei der Fahndung nach Drogenschmugglern, im Kampf gegen Korruption und Straßenkriminalität sind Computer unersetzliche Helfer geworden. Für Amigas Rolle im TV-Renner Miami Vice sprachen jedoch mehr die Fähigkeiten, dem pastellenen New-look-Touch des 45-Minuten-Clips gerecht zu werden.

Als am 9. April die deutschen Fernsehzuschauer dem Drogenfahnder Crocket bei seiner Arbeit zusehen durften, blickten sie für Sekunden auf einen Computermonitor, der offensichtlich ein Amiga-Window wiedergab.

Anfang 1986 gab es erste Gerüchte über Amigas Nebenrolle im Popspektakel Miami Vice. In der amerikanischen Amiga-Szene war niemand überrascht. Wenn es um futuristisch anmutende Hochglanzgrafiken geht, ist der Amiga unbestritten einsame Spitze. Das paßt gut in das Konzept von Miami Vice, optische und musikalische Elemente aus der aktuellen Kultur- und Konsumszene zu einer Krimiserie zusammenzufügen — Zeitgeist eben. Wenn Don Johnson strumpflos in weißen Espandrilles durch surrealistische Bildkompositionen schlappt, finden sich reichlich Zitate an aktuelle Kunstströmungen. In dieses Konzept fügt sich der „Maserati unter den Homecomputern“ hervorragend ein, während der unkonventionelle Star einen schwarzen Ferrari kutschiert — so blank poliert, daß man auf der Motorhaube die Kokusnüsse der sich spiegelnden Palmen zählen kann.

Am Drehort von Miami Vice hatte Vinoy Laughner, freier Autor für Amiga World, Gelegenheit, mit Bob Lacey, dem Chefdekorateur des Filmteams zu sprechen. Er ist verantwortlich für das Outfit der Sendung. Jedes Möbelstück, Poster und eben auch jeder Computer wird von ihm ausgewählt und plaziert. Auf der Suche nach neuen Eindrücken und Farben für die nächste Seriensaison (mehr Pink und Aquamarin) fiel ihm der Amiga auf. „Wir werden in der Vice-Station einen neuen Raum einrichten, voll mit hochentwickelten technischen Geräten für die Beamten. Hier kann der Amiga ebenso plaziert werden wie im Überwachungswagen.“ Die Crew nennt diesen Wagen „Bug Van“, Wanzenauto — sicher nicht nur wegen der Tarnung als Lieferwagen eines Kammerjägers mit einem riesigen Insekt auf dem Dach. „Der Amiga könnte digitalisierte Portraits des Gaststars der Woche als Auszug aus der Verbrecherkartei oder aufbereitete Daten von Überwachungskameras und Abhöranlagen präsentieren.“

Cooler Werbeeffekt

Man kann über Miami Vice geteilter Meinung sein. Viele werden es „echt cool“ finden, andere fragen sich, warum sich Don Johnson nie rasiert. Ganz sicher ist ein Alligator auf einem Hausboot keine alltägliche Erscheinung in Florida, und was haben die Sehnsüchte nach Sommer, Sonne, Palmen und Meer mit einem Computer zu schaffen. Nun ist es doch nett, wenn man seinen Lieblingscomputer in einer bekannten Sendung wiedersieht. Beweist es doch, daß der Amiga als Vorhut einer neuen Computergeneration anerkannt ist, denn Miami Vice ist als reine High-Tech-Show konzipiert, die sich nur mit brandaktuellen und hochwertigen Dingen abgibt. Genauso denken auch die User über ihren Computer.

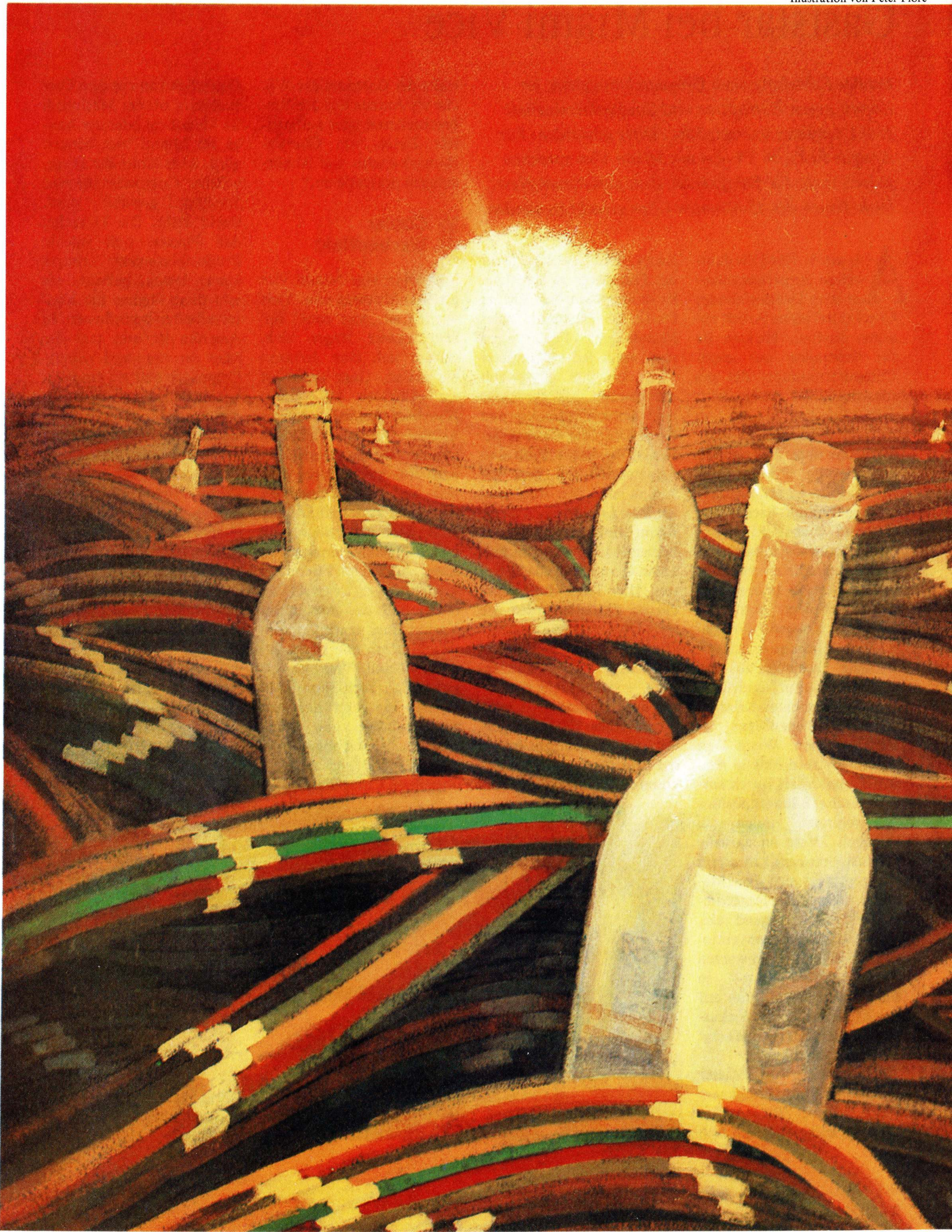
Eigentlich kann es dem User egal sein, ob der Macintosh im Mondlicht oder der Amiga bei Miami Vice benutzt wird. Aber hinter der Frage, welcher Computer für eine Sendung gewählt wird, steckt mehr. Die Tatsache, daß Firmen sich solches Productplacement einiges kosten lassen, beweist, wie sehr diese Medien das Käuferverhalten beeinflussen. In Amerika ist nun mal der Mac der Standard-Yuppie-Computer und ist des öfteren in Serien zu sehen, die die Probleme junger Aufsteiger behandeln. Wie gut harmoniert da der Amiga mit einer Krimiserie, die von Glitter, High-Tech, schrillen Farben, fetziger Musik und atemberaubenden Kamerafahrten lebt. Vielleicht sollte Commodore auf der neuen Extradiskette die Parameter der Translate-Bibliothek so ändern, daß der Say-Befehl mit der Stimme von Don Johnson spricht: „Okay, Pal, die Party ist vorbei. Gib auf, bevor ich dich in Haifischfutter verwandele. . .“

(Vinoy Laughner/ deutsch: uk)



Eine Stadt in Pink und Aquamarin: der passende Drehort für Amigas Gastrolle

Illustration von Peter Fiore



Hallo Amerika!

Falsch verbunden ist man mit amerikanischen Netzwerken nie. Dort findet der User eine Fülle von Informationen rund um den Amiga, die normalerweise nur mit großer Verzögerung Deutschland erreichen.

Es ist gar nicht so schwer in den Genuß der amerikanischen Daten-, Informations- und Programmflut zu gelangen. Drei Dinge sind dazu nötig: ein Amiga, Modem und Zugang zu einem Netzwerk. Jetzt kann man dann die bekanntesten Mailboxen direkt anwählen und sich als Benutzer eintragen. Nach einiger Zeit trudelt dann ein Antragsformular ins Haus. Ausgefüllt und abgeschickt: Systeme wie BIX oder CompuServe zählen nun einen Benutzer mehr. Da heißt es aufpassen — denn allzuschnell ergreift die Sucht den Neuankömmling. Konferenzen, Programme und Kontakte zu anderen Amigabenutzern — schier endlos sind die neuen Möglichkeiten, der heimischen Datenmühle Neuigkeiten zu entlocken. Dann verursachen am Monatsende nicht nur die Deutsche Bundespost, sondern auch die Mailboxen mit ihren Benutzungsgebühren schnell einen familiären Börsenkrach. Billig ist das wirklich nicht. Deutschland liegt bekanntlich mit seinen Preisen für Gebühreneinheiten ganz ungebührlich weit vorne in der Weltrangliste. Wem das Geld für solche Aktivitäten zu schade ist, erfährt in diesem Artikel zumindest, was drüben alles möglich ist. Denn bevor derartige Informationssysteme nach Deutschland kommen, wird wohl noch einiges dreckiges Wasser den Rhein runter fließen. Natürlich kann AmigaWelt hier nur eine kleine Auswahl der bekannten Informationssysteme beleuchten. Wir haben aber die für Amiga-User interessantesten herausgepickt. Im Mittelpunkt stehen dabei die kommerziellen Informationssysteme, für die der

Von hüben nach drüben: DFÜ als Reiseticket

hierzulande übliche Begriff „Mailbox" schon fast eine Beleidigung ist. Diese Systeme sind oft sehr komplex aufgebaut und die Bedienung mit allen Feinheiten ist nur den erfahrenen Spezialisten geläufig. Doch auch mit wenig Kenntnis lassen sich Briefe lesen und versenden, Dateien überspielen oder Konferenzen abhalten. Im folgenden werden die besonderen Eigenschaften eines jeden Systems vorgestellt — andere Daten, wie Baudraten, Preise und Mitgliedsbeiträge, sind in dem Übersichtskasten zusammengefaßt.

Für alle: „American People Link"

„Plink" — ein Kürzel, das für „American People Link" steht, was soviel heißt wie „Verbindung der amerikanischen Bevölkerung". Das ist bei weitem nicht zu hoch gegriffen, denn hierbei handelt es sich um eines der größten Amiga-Systeme überhaupt. Schon seit einigen Jahren etabliert, hat sich Plink zu einem Knotenpunkt für Informationen, Programme und Neuigkeiten gemausert. Die getrennten Bereiche für Briefe, Programme und Konferenzen sind durch die wahlweise Menü- oder kommandogesteuerte Benutzung sehr einfach zu durchforsten, und man ist oft schneller als einem lieb ist in Diskussionen mit anderen Teilnehmern verwickelt. Vor allem die sehr niedrigen Kosten — umgerechnet unter 5 DM pro Stunde Benutzung — machen Plink zu einem beliebten Treffpunkt der „normalen" Benutzer, was sich durch eine große Mitgliederzahl ausdrückt. Alle Meldungen sind neuesten Datums. Hunderte von veralteten Meldungen werden jede Woche aus dem System entfernt, um belegten Speicher wieder frei zu machen. Neben selbstverständlichen Abteilungen für Grafik und Programmierung gibt es Spiel- und Musiksektoren um eine Fachsimpelei zu beginnen. Pro Woche erscheinen in den Programmsektoren mehr neue Public-Domain-Programme, als man sich überhaupt in vernünftiger Zeit übertragen lassen kann. Das liegt vor allem daran, daß die Überspielzeit von Programmen nicht als Benutzerzeit gerechnet wird und keine Benutzergebühren kostet. Größenordnungen von mehreren hundert angebotenen Amiga-Programmen sind bei Plink keine Seltenheit. Das Spektrum der Programme reicht dabei von Programmierhilfen bis zu Demonstrationen und kompletten Anwendungspaketen, die oft besser sind als die kommerziellen Produkte aus diesem Bereich. Erfreulicherweise wurde bei Plink ▶

Amiga America
001/619/364-3816
SYSOP: Chet Solace

AMIC
001/707/579-0523
SYSOP: Dave Salas
Dave ist der Autor des bekannten Amiga Terminal Programms AMIC

F.A.U.G.
001/415/595-2479
First Amiga User Group
Ein sehr populärer Service



ein neuartiges Übertragungsprotokoll integriert, „Windowed XModem", das auf dem bekannten XModem basiert. Dateinamen werden auf Wunsch automatisch mitübertragen und die Transfergeschwindigkeit wurde bei gleichbleibender Baudrate erhöht, weshalb WXModem jetzt zu den schnellsten Protokollen überhaupt zählt. Trotzdem ist WXModem noch sehr sicher und wird sich eventuell schon bald als neuer Standard bei Amiga- „Mailboxen" durchsetzen. Leider gibt es aber auch bei Plink einen entscheidenden Nachteil. Denn trotz der verschiedenartigen Benutzerführungen mit Menüs und/ oder Kommandos bleiben „dank" des spärlichen Benutzerhandbuchs, das mit der Registrierung versandt wird, noch viele Fragen offen, die sich meist nur durch Ausprobieren klären lassen.

Noch mehr Informationen: „GEnie"

Wer ein System sucht, welches sich in seinen Aktivitäten nicht nur auf Computer und technische Aspekte bezieht, wird an GEnie Freude haben. GEnie bietet wesentlich mehr Themenbereiche an als Plink. Die Spanne reicht von obligatorischen Dingen, wie Grafik und Musik, über Börsenberichte, Neuheiten aus der Telekommunikations-Industrie bis hin zu Familienkonferenzen. Sogar eine amerikanische Fluggesellschaft ist in diesem Rechnersystem vertreten. Spezielle Funktionen erlauben die sehr effektive Suche nach bestimmten Stichwörtern in allen neuen Meldungen des Systems. So können aus dem Überangebot an Mitteilungen und Eintragungen in die öffentlichen Sektionen, genau die herausgefiltert werden, die das Wort „Terminal" enthalten. Auf der Suche nach einem neuen DFÜ-Programm ein sehr nützliches Hilfsmittel. Ein weiterer Service erfreut vor allem die Eltern kleinerer Kinder. Mit „Grolier's Encyclopedia" steht ein leistungsfähiger Bereich zur Verfügung, der bei den Schulaufgaben helfen soll. „Amiga Roundtable" nennt sich der wohl interessanteste Teil des Systems. Für unter 10 DM pro Stunde (im 1200 Baud Zugriff) können aus dem reichhaltigen Programmangebot zahlreiche Lösungen für eventuelle (Programmier-)Probleme gefunden werden. Als Protokoll findet hier das weitverbreitete XModem Verwendung, welches — über spezielle Telefonleitungen benutzt — nur geringfügig langsamer als das WXModem von Plink ist. Auch der Umfang des Programmangebotes läßt sich mit Plink vergleichen. In jedem Fall bieten beide Systeme normalerweise weit mehr Informationen, als ein einzelner Mensch sinnvoll verarbeiten kann. Der eigentliche Knüller des GEnie-Systems ist eine komplette Flugsimulation für Amiga- und Macintosh-Benutzer. Um in den Genuß dieses Multi-User-Spiels zu kommen, muß allerdings erst die komplette Software geladen werden. Drei und vier Stunden kann das dauern. Danach läßt sich damit zu Hause üben, damit man nicht beim ersten Kampf sofort abgeschossen wird. „Air Warrior" — so der Name des Programmes — ist eine Kampfflugsimulation des 2. Weltkriegs, die den „DogFight"-Modus des Amiga-„Fligh Simulator II" von SubLogic an Attraktivität weit übertrifft. Nicht nur daß aus mehreren, bekannten Flugzeugtypen des 2. Weltkriegs ausgewählt werden kann — nein, auch Geräusche und Grafiken (die man mit dem Terminalprogramm erhält und nur noch über DFÜ abgerufen werden) sind wesentlich besser gelungen. Man ist mit Landkarten, wie auch mit einem reichhaltigen Waffenarsenal ausgestattet. Clou des Spiels ist natürlich die Möglichkeit, gegen Mitspieler antreten zu können, die sich irgendwo in der Welt befinden. Zur Erscheinungszeit dieses Artikels wird auf GEnie höchstwahrscheinlich ein weiteres Spiel laufen, daß die — in den USA sehr bekannte — Dungeons-and-Dragons-Sparte vertritt.

Abenteuer von Kontinent zu Kontinent

Dies sind Spiele, die sehr auf taktisches Überlegen setzen, Textadventures nicht unähnlich und mit sehr vielen Leuten gleichzeitg gespielt werden. In einem Großrechnersystem machen solche Spiele, vergleichbar mit „Bard's Tale" oder „Ultima", wesentlich mehr Spaß, als im stillen Kämmerlein. Dieses „GemStone" genannte Programm wurde vom Programmierer Dave Whatley auf Amiga entwickelt.
Das erste und bisher wohl auch eines der größten Systeme für die Öffentlichkeit ist CompuServe. Die Möglichkeiten und Angebote dieses Rechners in einem ▶

Amerikanische Informationssysteme und ihre Preise

Auch in den USA wird im Telefonnetz Tag- und Nachttarif unterschieden. Der Tagtarif (TT) geht meist von 7 Uhr morgens bis 6 Uhr abends (jeweils dortige Uhrzeit). In der restlichen Zeit und an Sonn- und Feiertagen herrscht Nachttarif (NT). Die Sparte „Erreichbar über" stellt eine kurze Liste der Netzwerke dar, über die Sie das jeweilige System erreichen können. PC Pursuit stellt hier allerdings eine Ausnahme dar. Es ist kein eigenes Netzwerk, sondern ein spezieller Service von Telenet und nur der Vollständigkeit halber aufgeführt. Alle Netzwerke sind über den Datex-P-Dienst der Post erreichbar, über den man sich in jedem Postamt informieren kann. Diese Liste ist natürlich nur ein Auszug der wichtigsten Gebühren. Die Benutzungskosten sind dabei aus Gründen der Vereinfachung pro Stunde Benutzung angegeben, obwohl sie minütlich abgerechnet werden. Eine komplette Gebührenübersicht erhält man bei der Anmeldung als Benutzer.

Plink American PeopleLink

Nummer: 001-800/524-0100
Mitgliedsbeitrag: keiner
Erreichbar über: Telenet, Tymnet, PC Pursuit
Baudraten und Preise (stündlich):
NT: $4,95 (300/1200), $11 (2400)
TT: $12,95 (300/1200), $14,95 (2400)
Mit PC-Pursuit (Zusatzgebühr $25 pro Monat)
Nur NT: $3 (bis 2400)

BIX Byte Information Exchange

Nummer: 001-800/227-2983
Mitgliedsbeitrag: $39 (einmalig)
Erreichbar über: Tymnet
Baudraten und Preise (stündlich):
NT: $11 (bis 2400)
TT: $20 (bis 2400)

CIS CompuServe Information Service

Nummer: 001-800/848-8199
Mitgliedsbeitrag: keiner
Erreichbar über: direkte CIS-Leitungen (nur für die USA interessant), Telenet, Tymnet
Baudraten und Preise (stündlich):
NT & TT: $6 (300), $12,50 (1200)
plus $.25 pro Stunde für die CIS-Leitungen
plus $2 pro Stunde für Telenet- und Tymnet-Verbindungen (NT)
plus $6 pro Stunde für Telenet- und Tymnet-Verbindungen (TT)

GEnie General Electric Information Service

Nummer: 001-800/638-9636
Mitgliedsbeitrag: $29,95 (inklusive $10 Verbindungskosten)
Erreichbar über: spezielle GEnie-Leitungen (von Deutschland nur über sogenannte Outdials erreichbar)
Baudraten und Preise (stündlich):
NT: $5 (300/1200), $12,50 (2400)
TT: $35 (300/1200), $42,50 (2400)
plus $2 pro Stunde für einige Bereiche der USA

The Well Whole Earth Software Catalog

Nummer: 001-415/332-4335
Mitgliedsbeitrag: $8 pro Monat
Erreichbar über: Tymnet, PC Pursuit
Baudraten und Preise (stündlich):
NT: $5 (300/1200)
Mit PC-Pursuit (Zusatzgebühr $25 pro Monat)
NT: $3 (300/1200)

solchen Artikel zu erläutern, würde den Rahmen sprengen. Umfaßt es doch — normalerweise gegen entsprechende Aufpreise — neben aktuellsten Börsenberichten auch die Möglichkeit Bestellungen aufzugeben und fantastische Spiele. Mit fast 300 000 Benutzern bieten sich vielfältige Kontaktmöglichkeiten für vereinsamte oder vom Guru frustrierte Computerbenutzer. Unterteilung in thematische Foren erhöht die Wahrscheinlichkeit die richtigen Informationen und Leute an den Draht zu bekommen. Leider ist jedoch gerade der Amiga schwächer vertreten als bei Plink oder GEnie. Das betrifft vor allem die Anzahl der erhältlichen Programme, die sich keineswegs mit dem immensen Angebot von Plink vergleichen läßt. Trotzdem — oder vielleicht gerade deswegen — bieten die öffentlichen Bretter in den Amigabereichen sehr viel Informationen von hohem technischen Niveau. Viele der hier enthaltenen Meldungen sind die hohen Benutzungskosten von über 20 DM pro Stunde — im Zugriff mit 1200 Baud — wert.

Tummelwiese für Profis: BIX

Wem diese Systeme nicht genug bieten, der findet vielleicht Gefallen an BIX. „Byte Information Exchange" ist die offizielle DFÜ-Adresse des bekannten, amerikanischen Magazins „Byte". Für Programmierer, technisch interessierte Amigabenutzer oder einfache Freaks gibt es kaum einen Platz, wo mehr Wünsche erfüllt werden können. Fast jede Person, die in der Amigaszene bekannt ist — von bekannten Public-Domain-Autoren bis hin zu Hardware-Entwicklern — ist in BIX vertreten.

Frage und Antwort über BIX

Trotz der hohen Verbindungskosten — ähnlich wie bei CompuServe gut 20 DM pro Stunde — und dem Fehlen eines irgendwie gearteten Konferenzmodus ist BIX mit seinen vielfältigen Angeboten gerade im Amigabereich führend. Durch die zahlreich vertretenen Programmierer, die meist auch bereitwillig auf Fragen antworten, findet man hier Antworten auf viele, vor allem programmtechnische, Probleme. Es kann natürlich auch vorkommen, daß für eine Aufgabenstellung die komplette Programmlösung schon im Netz bereitliegt. Gerade wegen dieser riesigen Vielfalt an Programmen sollten die hohen Benutzungskosten niemanden schrecken. Außerdem lassen sich durch schnelles Einladen der Briefe in den Speicher mit anschließender Auswertung am heimischen Rechner sehr viel Kosten einsparen. Auch die Antworten können selbstverständlich offline, also ohne direkte Verbindung zum Rechner, vorgeschrieben und dann schnell an die entsprechenden Personen überspielt werden.

Größtenteils im PC-Bereich bekannt ist ein System namens „The Well". Es ist wesentlich kleiner als alle vorher angesprochenen Rechner und bietet deshalb auch nicht so große Benutzerzahlen. Trotzdem hat sich hier eine kleine Amiga-Fan-Gemeinde aus der Westküste Amerikas zusammengetan, um der PC-Übermacht Paroli zu bieten. Es gibt zwar nicht so riesige Mengen an Informationen, die Betreuung und der Kontakt zu anderen Usern ist jedoch persönlicher: Man hat mehr Zeit füreinander. Mit einem eigenen Zugang zum „UseNet"-Verbund gleichen mehrere Rechner des Systems den Quantitätsnachteil aus (siehe AmigaWelt 2/88 „Weltweites Netz gespannt"). Als Nachteil ist zu vermerken, daß hier wie bei BIX echte Konferenzen fehlen.

Wie in allen Bereichen des Lebens haben große Sachen meist auch große Haken. Genauso verhält es sich mit obigen Informationssystemen, die zwar eine Fülle von Nachrichten zur Verfügung stellen, aber nicht gerade leicht zu bedienen sind. Mit einer echten, reinen Menüsteuerung kommt kein System dieser Größe mehr aus. So werden Anfänger bestimmt Probleme mit der Bedienung haben. Zur Einübung bieten Mailboxen oder BBS (Bulletin Board Service) gute Gelegenheiten. Diese sind nämlich, bis auf den Umfang der Meldungen, der Benutzer und der Programmvielfalt, meist ein Abbild der großen Systeme. Bieten aber den Vorteil, daß die Benutzung meist kostenlos ist und der Zugriff jedem gestattet wird, der im Besitz der Telefonnummer ist. Solche Mailboxen gibt es auch in Deutschland. In USA existieren weit über hundert computerisierte „Briefkästen". Über Userclubs oder Freunde (auch bei den Fachhändlern finden sich immer wieder „Spezialisten"), läßt sich leicht die Nummer einer Mailbox herausfinden, in der fast immer aktuelle Listen von anderen Boxen in der ganzen Bundesrepublik bereitliegen.

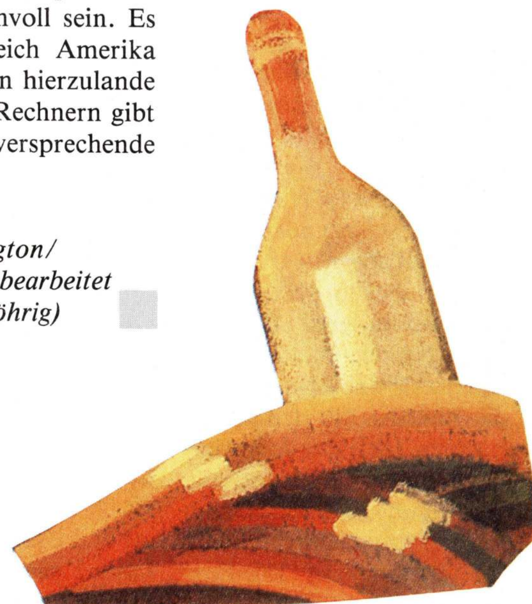
Und die Moral von der Geschicht'...

...ohne Modem geht es nicht! Das eigene Modem erweist sich bald als wichtigste und interessanteste Erweiterung für einen Computer überhaupt. Damit stehen alle Türen zu aktuellsten Informationen, fantastischen Programmen, umwerfenden Spielen und neuen Freunden offen. Leider schiebt zur Zeit noch die Deutsche Bundespost einen Riegel vor die „Information für alle". Durch die langsame Bearbeitung von FZZ-Zulassungen für neue Modems, versucht die Post mit einer Hinhaltetaktik ihre Modems zu überteuerten Gebühren an den Mann zu bringen und den weltweit etablierten Hayes-Standard für Modembefehle zu ignorieren. Bis zur Einlösung des Versprechens von Herrn Schwarz-Schilling, alle auf dem Markt befindlichen Modems auch anschließen zu dürfen, ist es wohl noch lange hin. Wichtigstes Kriterium beim Modemkauf ist die Baudrate. Mit den altertümlichen 300 Baud sollte sich niemand mehr zufriedengeben. Bei solchen „Geschwindigkeiten" ist nicht einmal das Träumen von irgendwelchen „Downloads" erlaubt. Der Zeitaufwand steigt ins Unermeßliche und außer der Telefon-

rechnung leidet auch die Geduld der anderen Benutzer, die sich auch mal „einloggen“, in den Rechner einsteigen möchten. Das Minimum für einen einigermaßen zukunftssicheren Kauf sollte daher bei 1200 Baud (vollduplex, gleichzeitige Datenübertragung in beide Richtungen) liegen. Dies hat sich in den USA fast als Standard durchgesetzt und wird auch in Deutschland alles Langsamere verdrängen. Mit einem 2400-Baud-Modem sind Sie dann natürlich für alle Eventualitäten gerüstet. Natürlich „vertragen“ die jeweils größeren Modems auch alle geringeren Geschwindigkeiten, so daß eine volle Kompatibilität gewährleistet ist. Keine Kopfschmerzen muß man sich über die Bedienbarkeit machen. Unter den bekannten und preisgünstigen Modems hat sich der Hayes-Kommandosatz als Standard herausgestellt und kein Modem ohne diese Unterstützung hat heute noch Marktchancen. Die meisten Telekommunikationsprogramme unterstützen diesen Befehlssatz dann sogar aus ihren Menüs heraus. Vertrackter wird die Sache allerdings bei den, meist sehr nützlichen, Spielereien, wie verschiedenen bunten Anzeige-LEDs, Mithöreinrichtungen und anderen Dingen.

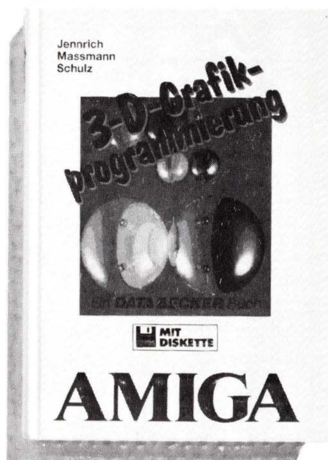
Die Verbindung mit Gleichgesinnten macht Spaß und kann sehr sinnvoll sein. Es muß nicht gleich Amerika sein. Unter den hierzulande angesiedelten Rechnern gibt es recht vielversprechende Ansätze.

(Peggy Herrington/ Übersetzt und bearbeitet von Ottmar Röhrig)



Illusion von Raum und Licht

Mit diesem Data-Becker-Buch wollen die Autoren Jennrich, Massmann und Schulz nicht nur eine komplette Lösung der computerberechneten Grafik präsentieren, sondern auch intensiv auf das Ray-Tracing sowie die dafür notwendigen mathematischen Grundlagen und die Entwicklung aller notwendigen Algorithmen eingehen. In acht Kapiteln beschreiben die Autoren die Eingabe der dreidimensionalen Objekte, Materialbestimmung, automatische Berechnung, die verschiedenen Lichtquellen und die Handhabung des IFF-Formats. Die beiliegende Diskette enthält alle Routinen als editierbare Basic-Versionen wie auch die compilierten Fassungen.



3-D-Grafikprogrammierung, Jennrich/Massmann/Schulz, Data Becker GmbH, Düsseldorf, 284 Seiten, 59 Mark.

Grundlagen und Tricks

Die im Düsseldorfer Sybex-Verlag erschienene Ausgabe „Amiga 500/1000" wendet sich an alle Amiga-Einsteiger. Der Autor, Thomas Binzinger, beschreibt im ersten Kapitel das Arbeiten mit der Workbench und geht dann zum Anschluß



zusätzlicher Geräte wie Fremdmonitore, Fernseher oder Stereoanlage über. Darauf folgt ein Einstieg in die mitgelieferten Programme und Utilities. Das sechste Kapitel geht auf die Hardware-Struktur des Amiga ein und behandelt das Betriebssystem, Grafik, Ton, Darstellungsarten und den Umgang mit Farbpaletten und -registern. Kapitel 7 widmet sich AmigaBasic und anderen Programmiersprachen. Wer sich für die Schnittstellen des Amiga interessiert, wird im zehnten Kapitel des Buches fündig. Im Anschluß daran folgen Tips und Tricks.

Amiga 500/1000 — Das Einsteigerbuch, Thomas Binzinger, Sybex-Verlag, Düsseldorf, 279 Seiten, 29,80 Mark.

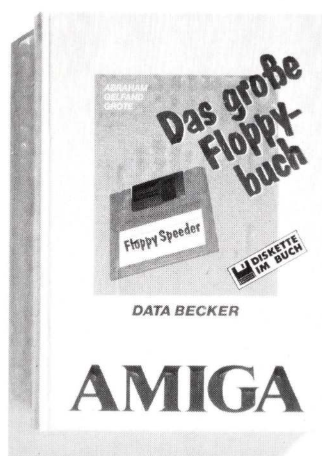
Einsicht in die Floppy

„Das große Floppybuch" spricht Anfänger und Fortgeschrittene an. Zunächst werden die Floppy-Funktionen der Workbench unter die Lupe genommen. Da lernen Neulinge, wie man Disketten kopiert, Dateien löscht und eine RAM-Disk installiert. Einfache Befehle wie „Install", „Protect", „Diskdoctor" oder „Addbuffers" veranschaulichen Diskettenoperationen mit dem CLI. Im folgenden Kapitel kann anhand umfangreicher (Basic-)Dateiprogramme geübt werden, wie eine sequenzielle Datei erstellt und erweitert oder wie direkt auf Dateien zugegriffen wird. Zum Thema Betriebssystem können selbst Fortgeschrittene etwas dazulernen. Listen aller Funktionen und Programmbeispiele in C schaffen Durchblick bei allen zu übergebenden Parametern und deren Aufgaben. In einer Analyse der File-Struktur wird klar, wie Diskettenblöcke aufgebaut und zusammengehörende Blöcke verknüpft sind. Zusätzlich wird die Handhabung des Diskettenmonitors erläutert. Dazu fehlen allerdings auf der mitgelieferten Diskette Quellcode und assembliertes Programm. Das hat zur bitteren Folge, daß im Bedarfsfalle 21 Seiten Listing abgetippt werden müssen.

Kernstück aller Floppy-Operationen ist das Trackdisk-Device, das für Lesen und Schreiben von Byte auf Diskette zuständig ist. Wie es arbeitet, zeigen gut dokumentierte Auszüge aus dem Betriebssystem. Allerdings versteht man die Funktionsweise nur auf der Basis fundierter Kenntnisse des 68000er-Befehlssatzes und der Assemblerprogrammierung.

Sehr ausführlich wird auf den Diskettenzugriff ohne DOS eingegangen. Der Leser erfährt alles über Aufzeichnungsformate, Codierung und Decodierung der Daten. Hervorragend dokumentierte, in Assembler geschriebene Programme tragen zum besseren Verständnis bei. Dem Buch liegt eine Utility-Diskette bei, die eins der besten Kopierprogramme, die zur Zeit verfügbar sind, enthält. Damit läßt sich sowohl ungeschützte als auch geschützte Software kopieren. Zusätzlich erhält der User einen in Assembler programmierten Floppy-Speeder. Lediglich der im Buch besprochene Diskettenmonitor fehlt auf der Diskette.

(Stephan Quinkertz)



Das große Floppybuch, Abraham/Gelfand/Grote, Data Becker, Düsseldorf, 398 Seiten, 1 Utility-Diskette, 59 Mark

IBM-Report

Vielen läuft bei den Buchstaben IBM ein kühler Schauer über den Rücken: Big Blue, wie der größte Computerkonzern der Welt genannt wird, steht für Macht, Technik und große Geschäfte. Stephanie Sand hat sich in ihrem Heyne Report „IBM — Eine kritische Geschichte des Computer-Giganten" nicht gescheut, hinter die Kulissen zu schauen und in den verfügbaren Quellen zu schnüffeln. Was sie zu Tage fördert ist sowohl spannend als auch informativ. Es ist die Geschichte eines Unternehmens, das mit Tabelliermaschinen anfing, und nun, mit 400 000 Mitarbeitern in 130 Ländern, über 51 Milliarden Dollar Umsatz macht. Zum Vergleich: Das nächstgrößte Computerunternehmen setzt „nur" 7,6 Milliarden Dollar um.

Wie IBM die Macht errang, und wie verbissen sie verteidigt wird, schildert Stephanie Sand auf 220 Seiten. Angefangen von der Lebensgeschichte des nahezu uneingeschränkten IBM-Diktators Thoma J. Watson bis zur genauen Durchleuchtung der vielen Prozesse schildert sie genüßlich, wie wirtschaftliche Macht mit nach ihrer Meinung unsauberen



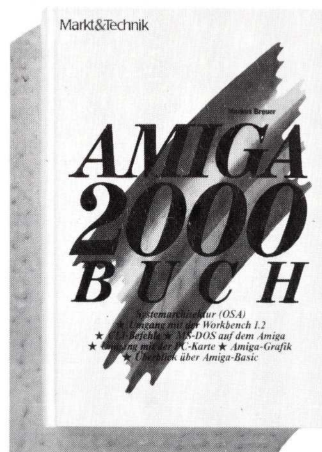
Methoden errungen und verteidigt wurde. Immer wieder klagten sie Konkurrenten und die Regierung der Vereinigten Staaten wegen illegaler Geschäftsmethoden an. Aber nur ein einziges Mal unterlag IBM.
Im Prozeß „United States vs. IBM" mußte das Gericht 46 726 Tonnen IBM-Unterlagen, das sind 66 Millionen Seiten Papier, zu den Akten nehmen und 2500 Zeugen anhören. 100 IBM-Juristen versuchten, den Vorwurf, IBM nutze seine marktbeherrschende Stellung ungesetzlich aus, zu widerlegen. Am 8. Januar 1982 gab die Regierung mit der Einsicht auf, daß ihre Vorwürfe gegen die IBM „jeder Grundlage" entbehrten.
Die Autorin lockert ihr mit knallharten Daten gefülltes Buch durch amüsante Anekdoten auf. So ist die Lektüre unterhaltsam und informativ zugleich. Wer sich für eine kritische Analyse des Blauen Riesen interessiert, den werden die 9,80 Mark nicht reuen.

(Peter Vogel)

IBM-Eine kritische Geschichte des Computer-Giganten, Stephanie Sand, Heyne Verlag München, 220 Seiten, 1988; 9,80 Mark

Tips für Experten

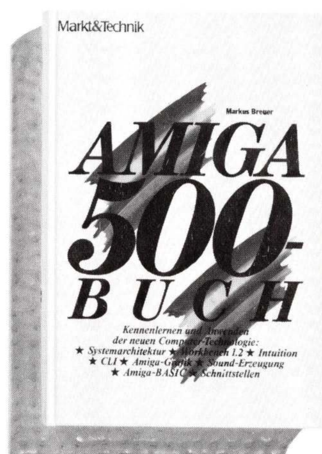
„Nicht für Anfänger", warnt Software-Profi Markus Breuer in der Einleitung seines Amiga-2000-Buches. Wie schon in seinem Handbuch für das Modell 500 stehen Expertentips im Vordergrund. Für Spielereien bleibt kein Raum. Nach der sehr ausführlichen Einleitung (Motto: „Wie war das noch gleich?") werden Workbench-Werkzeuge wie „Preferences" und Icon-Editor unter die Lupe genommen. Im nächsten Teil folgen Hinweise für den optimalen Einsatz des Command-Line-Interpreters und seiner Standardkommandos, bevor der Leser in den Ausbau des offenen Systems eingewiesen wird. Besitzern einer PC-Steckkarte sind 50 Seiten über die allgegenwärtige MS-DOS-Welt gewidmet. In den letzten Passagen kommt die Systemhardware zur Sprache (Festplatte, Spezial-Chips).



Amiga-2000-Buch, Markus Breuer, Markt & Technik Verlag AG, Haar bei München, 1988, 570 Seiten, 59 Mark.

Hinweise für den täglichen Umgang

Das Amiga-500-Buch wendet sich vor allem an ernsthafte Benutzer des kleinsten Vertreters der Amiga-Familie. Markus Breuer, selbst professioneller Programmierer, will nicht in erster Linie die phantastischen Fähigkeiten

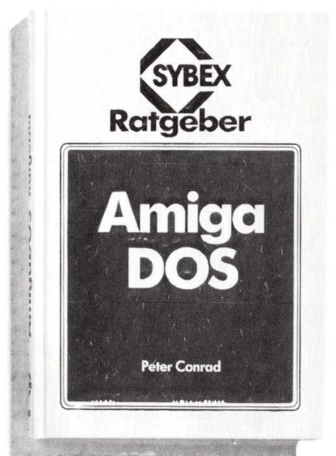


der Grafik- und Sound-Maschine demonstrieren, sondern bietet Hinweise für den täglichen Umgang mit Hard- und Software. Im Gegensatz zum „Amiga-Handbuch" (im gleichen Verlag erschienen) erfährt der Leser von der erweiterten Workbench 1.2 und anderen Besonderheiten des 500ers. Vom ersten Handgriff über Softwaretools, Projekte und die zeichenorientierte Benutzerschnittstelle bis zu Grafik- und Sound-Features behandelt der Autor auf knapp 500 Seiten alles Wissenswerte über Commodores handlichsten 68 000er.

Amiga-500-Buch, Markus Breuer, Markt & Technik Verlag AG, Haar bei München, 1987, 489 Seiten, 49 Mark.

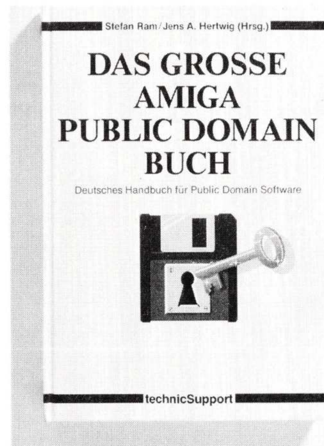
DOS-Ratgeber

Ein Handbuch kann der neue Sybex-Ratgeber „Amiga DOS" natürlich nicht ersetzen. Er dient vor allem als Nachschlagewerk für vergeßliche Profis und flinke Anfänger.
Das 146 Seiten starke Buch des Düsseldorfer Verlages ist in zwei Hauptthemen gegliedert. Der erste Teil führt kurz in das Betriebssystem AmigaDOS der Modelle 500, 1000 und 2000 ein und macht bekannt mit der Benutzeroberfläche und den allgemeinen Regeln im Umgang mit dem Kommandointerpreter. Teil II enthält die alphabetisch geordnete Befehlreferenz, in der die einzelnen Befehle und ihre Systax angegeben sind. Zusätzlich erläutert Autor Peter Conrad die von Commodore bisher dokumentierten Kommandos. Bei den Befehlreferenzen verwendete Beispiele orientieren sich an der AmigaDOS-Version 1.2 und der Workbench-Version 33.53.



Amiga DOS, Peter Conrad, Sybex Verlag, Düsseldorf, 1987, 146 Seiten, 29,80 Mark.

Schlüssel zu PD



Public Domain Software gibt's an jeder Ecke. Da fängt auch schon das Problem an. Es fehlt an Orientierungsmöglichkeiten im unüberschaubaren Angebot, es mangelt an Dokumentationen. Stefan Ram und Jens A. Hertwig bereiten diesem Zustand ein Ende. Im „Großen Amiga Public Domain Buch" wird PD-Software gegen Freeware und Shareware abgegrenzt. Dann geht es mitten in die Praxis. Zur Einführung werden CLI-Befehle, Bildschirmeditor Ed, Sicherheitskopien und Programmmanipulationen erklärt. Der erste Abschnitt des Hauptteils befaßt sich mit nützlichen Programmen wie Textverarbeitungen, Simulationen und Datentransfer-Software. Übersichtlich werden Zweck, Aufruf, Optionen und Menüs der Programme beschrieben. Zusätzlich wird auf Programme gleichen Anwendungsbereichs verwiesen. Bei der Hälfte aller Besprechungen ist die Adresse der Programmierer angegeben. So können interessierte PD-User programmspezifische Feinheiten erfahren. Grafiksoftware, Arbeits- und Programmierhilfen komplettieren die Software-Palette. Auf über 80 Seiten sind alle PD-Disketten der Serien „fishdisk", „faugdisk" und „panorama" aufgeführt. Die Listen sind alphabetisch, systematisch und nach Seriennummern geordnet. Herausgeber und Autoren wissen, wo der PD-Schuh drückt.

(ub)

„Das große Amiga Public Domain Buch", Stefan Ram, Jens A. Hertwig (Hrsg.), technicSupport GmbH, Berlin-(West) 1988, 348 Seiten, 49 Mark

200 Amigas in Atlanta

Anläßlich der Comdex-Messe in Atlanta präsentierte Commodore eine Amiga-Galerie: An 200 Rechnern konnte das Neueste und Beste aus der Amiga Hard- und Softwarepalette bestaunt werden.

Am Commodore-Stand waren Amiga 500 und 2000 die dominierenden Geräte: 25 Hersteller präsentierten ihre jüngsten Entwicklungen in den Bereichen Desktop-Presentation, Video-Animation, Utilities und Lernprogramme. Laut Commodore wurden weltweit bis jetzt 600 000 Amiga verkauft, 300 Soft- und Hardwareentwickler konzentrieren sich auf Commodores Freundin, über 1100 Programme sind bislang erhältlich. Zahlen, die beweisen, daß der Amiga auf dem Markt Fuß gefaßt hat.

„DR. Term" Professional heißt ein neues DFÜ-Programm von Progressive Peripherals. Seine besonderen Features sind ein intergriertes „Telefonbuch" mit 300 Einträgen und 40 per Funktionstasten aufrufbaren Makros (werden vom configuration file automatisch geladen). VT100, VT52 und TTY werden emuliert. Für Datenübertragung stehen xmodem, ymodem und wxmodem zur Verfügung. Mit DiskMaster kommt jetzt eines der umfangreichsten und leistungsfähigsten Utilities auf den Markt: Amiga-DOS-Kommandos werden per Mausklick ausgeführt. Zwei Directory-Displays stellen 70 File-Einträge dar. Zwölf physikalische oder logische Devices können im configuration-file definiert werden. IFF-Bilder und IFF-Sounds können abgerufen werden. Ausdrucken von Textfiles ist kein Problem. Für Mathematiker ist Math-Amation gedacht. Dieses Programm jongliert mit Formeln, stellt Grafiken und Wertetabellen dar. Integriert sind ein Scientific- und ein Matrix-Taschenrechner. Electronic Arts erweitert seine Delux-Serie und überrascht mit einer erfreulichen Meldung: Ab sofort werden die Delux-Programme nicht mehr mit Kopierschutz versehen und lassen sich so problemlos auf Harddisk installieren. DeluxePhotoLab macht Ausdruck und Manipulation von Bildern in Fotoqualität möglich, wie es bislang nur auf teuren Großrechnern möglich war. Das Programm besteht aus drei Teilen: Paint ist ein leistungsfähiges Malprogramm für alle Auflösungen. Die Größe des Zeichenbildschirms wird lediglich durch den verfügbaren RAM-Speicher begrenzt. Per Colors-Utility können Farben manipuliert, verändert und gemischt werden. Posters, das dritte Programm, erlaubt Ausdruck von Grafiken mit einer Größe von bis zu 10 x 10 Fuß. Deluxe Productions macht Business-Grafiken und Präsentationen zum Kinderspiel. 40 Spezialeffekte zum Ein- und Ausblenden von Grafik und Text machen das Programm auch zum Einsatz mit einem Genlock-Interface interessant. Im Video-Bereich tut sich einiges: Digitizer und Genlock-Interfaces werden besser und preiswerter. Wer die Anschaffung eines dieser Geräte plant, sollte sich deshalb etwas gedulden. AmigaWelt wird die neuen Produkte in den folgenden Ausgaben vorstellen.

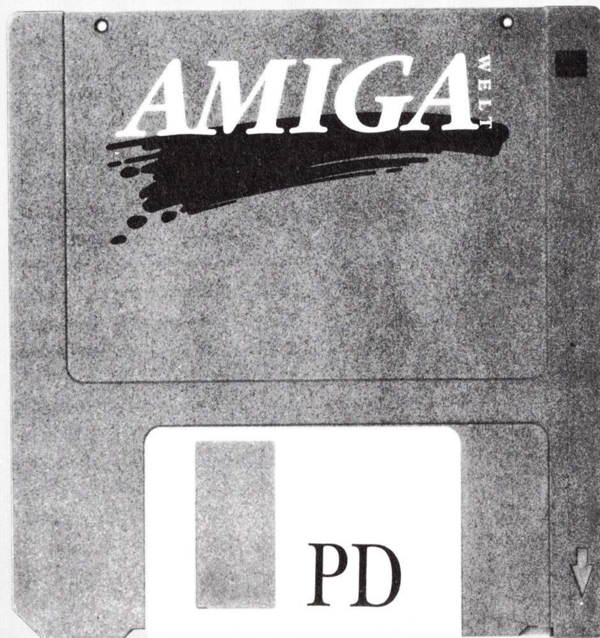
(mn)



Comdex '88: Am Commodore-Stand dreht sich alles um Amiga.

LESERSERVICE

Für nur 10 Mark plus 5 Mark Bearbeitungsgebühr und Versandkosten. Bei Nachnahme zusätzlich 5 Mark Gebührenpauschale!



Für alle,

die sich mühsames Abtippen, Falscheingaben, Testläufe und den Zeitaufwand sparen wollen, gibt es die Programme dieser AMIGA-WELT auf einer Diskette.

- **Patchen in Basic**
- **Makecat — Schneller Zugriff auf Programme**
- **Vom Amiga zum PC**
- **Icon Assembler**
- **Install — Abschaltung der Speichererweiterungen und Zusatzlaufwerke inklusive**
- **Color-Changer — Aufräumen in Farbpaletten**
- **Runback — Startkommando ohne CLI-Blockade**
- **Amiga-Kicks: nochmal die Programme aus 2/88**
- **Grafik in Bits und Bytes: einfaches Zeichenprogramm**

Diskette zu AMIGA WELT 3/88, Bestell-Nr. 805

Wir weisen darauf hin, daß die Anleitungen aus dem Heft sehr wichtig sind.

So können Sie bestellen: *Bitte vergessen Sie nicht, Ihre vollständige Adresse und die Bestellnummer anzugeben!*

Bei W. Kurtz
Postfach 26 01 51
8000 München 26

- ☐ *mit Scheck 15 Mark*
- ☐ *mit Postkarte (Nachnahme)*
- ☐ *auf das Postscheck-Kto. München, Kto. 43 60 36-803, 15 Mark einzahlen.*
- ☐ *telefonisch: 089/22 47 20 (Nachnahme)*

Obfuscated — Verwirrendes in C

84laman.c — *dritter Preis 1984: Mike Laman*
84mullender.c — *Hauptpreis 1984: Sjoerd Mullender & Robbert von Renesse*
85lycklama.c — *merkwürdigstes Programm 1985: Ed Lycklama*
86hague.c — *schlimmster Mißbrauch des C-Preprocessors 1986: Jim Hague*
86marshall.c — *bestes Layout 1986: Eric Marshall*
87biggar.c — *gelungenster Mißbrauch der Spielregeln 1987: Mark Biggar*
87hines.c — *miserabelster Stil 1987: Spencer Hines*
87westley.c — *bestes Layout 1987: Brian Westley*

Sie haben schon einen Blick auf die Listings geworfen? Es handelt sich nicht um die Gewinner eines Computergrafik-Wettbewerbs, wenngleich manches Programm durchaus die Bezeichnung „Kunstwerk“ verdient. ACHTUNG: Anfängern der C-Programmierung ▶

```
                                              extern int
                                                  errno
                                                    ;char
 argv, argc )                ;main(                 grrr
   r        ;               int    argc                r,
#define x  int i,         char *argv[];{int
x  ;if     (P(  !         j,cc[4];printf("                 P( );
&  P(j     )>2  ?         i                )   choo choo\n"  ) ;
;                         j                :      !  cc[  !    j ]
          for      (i=                     0;;    i  )[*  argv[i++ +!-i]
_exit(argv[argc- 2     / cc[1*argc]|-1<<4 ]   i++
  P  (    a  )    char a   ;  {     a  ;   while(  ) ;printf("%d",P(""));}}
 /* -    by E              ricM     arsh           a  >     "  B   "
                                                  all-      */;     }
```

```
#define DIT        (
#define DAH        )
#define __DAH      ++
#define DITDAH     *
#define DAHDIT     for
#define DIT_DAH    malloc
#define DAH_DIT    gets
#define _DAHDIT    char
_DAHDIT _DAH_[]="ETIANMSURWDKGOHVFaLaPJBXCYZQb54a3d2f16g7c8a90l?e'b.s;i,d:"
;main                      DIT                      DAH{_DAHDIT
DITDAH                     _DIT,DITDAH              DAH_,DITDAH DIT_,
DITDAH                     _DIT_,DITDAH             DIT_DAH DIT
DAH,DITDAH                 DAH_DIT DIT              DAH;DAHDIT
DIT _DIT=DIT_DAH           DIT 81                   DAH,DIT_=_DIT
__DAH;_DIT==DAH_DIT        DIT _DIT                 DAH;__DIT
__DIT'\n'DAH DAH           DAHDIT DIT               DAH_=_DIT;DITDAH
DAH_;__DIT                 DIT                      DITDAH
_DIT_?_DAH DIT             DITDAH                   DIT_ DAH:'?'DAH,__DIT
DIT''DAH,DAH_ __DAH        DAH DAHDIT               DIT
DITDAH _DIT_!=DIT          DIT_=2,_DIT_=_DAH_;      DITDAH _DIT_&&DIT
DITDAH _DAH_&223:DITDAH    DITDAH DAH_>='a'?        DITDAH
DITDAH                     DAH_ DAH DAH;            DIT
DITDAH DIT_+=              DIT_ DAH __DAH,_DIT_     __DAH DAH
DAH;}_DAH DIT DIT_         DIT DITDAH _DIT_>='a'?   DITDAH _DIT_-'a':0
DIT_>3?_DAH                DAH}                     __DIT DIT
DAH:'\0'DAH;return         DIT                      DIT_>>1
DIT_&1?'-':'.';}__DIT DIT                           DIT_ DAH _DAHDIT
DIT_;{DIT void DAH write DIT                        1,&DIT_,1 DAH;}
```

```
P;

a[900];        b;c;d=1
1)char*        *1;{g=
0;k*k<         g;b=k
g;++h);        --h;c=(
while(d        <=g){
;++f)a[        b<<5|c]
&&d<=g;        ++f)a[b
;}for(c        =0;c<h;
b){if(b        <k/2)a[
<<5|c]^=       a[b<<5
;printf(       a[b<<5|c
|c]);}         putchar(
```

```
#include <stdio.h>
#include <malloc.h>
main(togo,toog)
int togo;
char *toog[];
{char *ogto,   tgoo[80];FILE  *ogot;  int    oogt=0, ootg,  otog=79,
ottg=1;if (    togo==  ottg)   goto   gogo;  goto   goog;  ggot:
if (   fgets(  tgoo,   otog,   ogot)) goto   gtgo;  goto   gott;
gtot:  exit(); ogtg: ++oogt;   goto   ogoo;  togg:  if (   ootg > 0)
goto   oggt;   goto    ggot;   ogog:  if (   !ogot)  goto   gogo;
goto   ggto;   gtto:   printf( "%d    goto   \'s\n", oogt); goto
gtot;  oggt:   if (    !memcmp( ogto, "goto", 4)) goto  otgg;
goto   gooo;   gogo:   exit(   ottg); tggo:  ootg=   strlen(tgoo);
goto   tgog;   oogo: --ootg;   goto   togg;  gooo: ++ogto;   goto
oogo;  gott:   fclose( ogot);  goto   gtto;  otgg:   ogto=  ogto +3;
goto   ogtg;   tgog:   ootg-=4;goto   togg;  gtgo:   ogto=  tgoo;
goto   tggo;   ogoo:   ootg-=3;goto   gooo;  goog:   ogot=  fopen(
toog[  ottg],  "r");   goto    ogog;  ggto:   ogto=  tgoo;  goto
ggot;}
```

```
                 char rahc
                    [ ]
                     =
                   "\n/"
                     ,
                 redivider
                    [ ]
                     =
        "Able was I ere I saw elbA"
                     ,
                     *
              deliver,reviled
                     =
                    1+1
                     ,
                niam ; main
                    ( )
                   {/*\}
                    \*/
                  int tni
                     =
                    0x0
                     ,
              rahctup,putchar
                    ( )
           ,LACEDx0 = 0xDECAL,
                 rof ; for
              (;(int) (tni);)
                (int) (tni)
          = reviled ; deliver =
                 redivider
                     ;
for ((int)(tni)++,++reviled;reviled* *deliver;deliver++,++(int)(tni)) rof
                     =
             (int) -1- (tni)
           ;reviled--;--deliver;
              (tni)  =  (int)
          - 0xDECAL + LACEDx0 -
                 rof ; for
(reviled--,(int)--(tni);(int) (tni);(int)--(tni),--deliver)
             rahctup = putchar
            (reviled* *deliver)
                     ;
             rahctup * putchar
             ((char) * (rahc))
                     ;
                    /*\
                   {\*/}
```

```
short main[] = {
        277, 04735, -4129, 25, 0, 477, 1019, 0xbef, 0, 12800,
        -113, 21119, 0x52d7, -1006, -7151, 0, 0x4bc, 020004,
        14880, 10541, 2056, 04010, 4548, 3044, -6716, 0x9,
        4407, 6, 5568, 1, -30460, 0, 0x9, 5570, 512, -30419,
        0x7e82, 0760, 6, 0, 4, 02400, 15, 0, 4, 1280, 4, 0,
        4, 0, 0, 0, 0x8, 0, 4, 0, ',', 0, 12, 0, 4, 0, '#',
        0, 020, 0, 4, 0, 30, 0, 026, 0, 0x6176, 120, 25712,
        'p', 072163, 'r', 29303, 29801, 'e'
};
```

```
;e=1;f;
atoi(*
++>>1)
(h+=g>h
++0;for
=d++,b+=
<<5|c]=
++c){
b<<5|c]
|c]^=a[
]?"%-4d"
'\n');}}
```

```
g;h;O;
++1);
;for(h=
*(h+1))
(f=0;f<
e;for(
d++,c+=
for(b=0
^=a[(k
(k-(b+1
:"     "
/*Mike
```

```
main(k,
for(k=
0;h*h<=
-1)>>1;
O&&d<=g
f=0;f<O
e;e= -e
;b<k;++
-(b+1))
))<<5|c]
,a[b<<5
Laman*/
```

```
#define o define
#o ___o write
#o ooo (unsigned)
#o o_o_ 1
#o _o_ char
#o _oo goto
#o _oo_ read
#o o_o for
#o o_ main
#o o__ if
#o oo_ 0
#o _o(_,__,___)(void)___o(_,__,ooo(___))
#o __o
(o_o_<<((o_o_<<(o_o_<<o_o_))+(o_o_<<o_o_)))+(o_o_<<(o_o_<<(o_o_<<o_o_)))
o_(){_o_ _=oo_,__,___,____[__o];_oo ______;_____:___=__o-o_o_; _______:
_o(o_o_,____,__=(_-o_o_<___?_-o_o_:___));o_o(;__;_o(o_o_,"\b",o_o_),__--)
);
_o(o_o_," ",o_o_);o__(--___)_oo
_______;_o(o_o_,"\n",o_o_);______:o__(_=_oo_(
oo_,____,__o))_oo ______;}
```

wird dringend geraten, sich diesen Artikel für spätere Zeiten aufzusparen, da sie ansonsten ihren Glauben an das Gute im Menschen verlieren könnten. Sind wir jetzt unter uns? Sicherlich besitzt C leichte Tendenzen zum Kryptischen, aber gegenüber dem allseits verachteten BASIC sind C-Programme doch klar und deutlich, weil gar nicht so verschlungen und umständlich programmiert werden kann, wie etwa in BASIC. Wirklich? Na dann schnallen Sie sich mal an!

Normalerweise sollte die Qualität des Quelltextes mit der Erfahrung des Programmierers steigen, umständlichen und wenig eleganten Code wird man meist Anfängern zuordnen können. Für die folgenden Beispiele muß man wohl eine Ausnahme von dieser Regel machen: Nur wer C und seinen Compiler wirklich kennt, der kann so undurchschaubar programmieren.

The International Obfuscated C Code Contest

Alles fing, wie schon so oft, in UNIX-Kreisen an: So soll der Quelltext zur Bourne Shell (für Nicht-UNIX-Anwender, auch wenn ich für den Vergleich sicherlich gelyncht werde, sei hier gesagt, daß das so etwas ähnliches wie ein CLI ist) an einer Stelle so gräßlich gewesen sein, daß man sich dachte: Wenn DAS als vernünftige Programmierpraxis gepriesen wird, wie tief mag da erst das Niveau sinken, wenn man sich wirklich darum bemühte.

Es entstand ein Wettbewerb daraus. Seit 1984 wird jährlich in verschiedenen Kategorien das jeweils beste (oder auch schlechteste, je nach Standpunkt) C-Programm gewählt. Wie bei den Amerikanern so üblich, darf man das nicht mit tierischem Ernst sehen. Die gewählten Programme sollten eher als abschreckendes Beispiel dienen, denn so sollte kein C-Programm aussehen. Vom Unterhaltungswert abgesehen, kann man aus diesen Programmen natürlich vielerlei Feinheiten der Programmiersprache C erlernen. Auch wird so mancher Compiler (oder auch Lint) auf eine harte Probe gestellt, denn kaum eines der Programme geht zimperlich mit den Regeln der Sprache um.

Härtetest für Compiler: Jux-Programme nutzen jeden Winkelzug

Dementsprechend macht auch nicht jeder Compiler diese Prozedur mit. Bevor Sie also wie wild die Listings eintippen und dann nicht zum Laufen bekommen, lesen Sie lieber weiter. Die Regeln für das erste Jahr des Wettbewerbs, 1984, bestanden hauptsächlich in einer Begrenzung der Quelltextlänge auf 512 Bytes, und die meiste Verwirrung wurde daraus gewonnen, daß ungewöhnlich formatiert und der Preprocessor teils bis zum Exzeß genutzt wurde, nicht zuletzt der Längenbegrenzung wegen. Was dabei herauskam, sehen Sie an den beiden Listings jenes ersten Jahres. Um Programme wie das Mike Lamans zu verstehen, sollten Sie erst einmal versuchen, es auf die von Ihnen bevorzugte Weise (mit Einrückungen etc.) zu formatieren. Wichtig zu wissen ist, daß der Compiler Definitionen ohne Typ, wie in der ersten Zeile des genannten Listings, implizit als „int" annimmt, was auch für das Resultat und die Parameter von Funktionen gilt, wenn diese nicht näher spezifiziert werden. Jedoch sollte es auch ohne Änderungen am Programm nach Eingabe von ...

laman 25

... rundgehen. Probieren Sie den Aufruf auch mit anderen positiven Zahlen. Schwieriger wird es da schon beim zweiten Programm, dem Hauptpreis des Jahres 1984, das auf dem Amiga gar nicht erst läuft, da es maschinenabhängig für die VAX-11 oder PDP 11 entwickelt wurde. Wie kann es sich hierbei überhaupt um ein lauffähiges Programm handeln? Die Werte des Feldes „main []" repräsentieren ein kleines Maschinenprogramm, das vom normalen C-Startup-Code aufgerufen wird. Nach den Startvorbereitungen wird also direkt in den Datenbereich gesprungen. Da dies in den meisten Fällen weder Linker noch Loader stört, entsteht daraus ein lauffähiges Programm. Etwas entfernt ähnliches läßt sich an vielen Beispielprogrammen zum Modula-Compiler-M2Amiga oder auch dem neuen Install-Befehl in diesem Heft beobachten, wenn versucht wird, Assemblercode mit einem Compiler zu bearbeiten, der dafür nicht vorgesehen ist. Um weitere abartige Programme dieser Sorte zu verhindern, wurden die Regeln für die darauffolgenden Jahre derart abgeändert, daß ein Programm nicht mehr rechnerspezifisch sein durfte. Weitere Bosheiten bestanden darin, daß E1 [E2] per definitionem gleich (★((E1) + (E2))) und demnach auch gleich E2 [E1] ist. Daraus entstanden dann Ausdrücke in der Art von ...

i [„Hello worldNn"]

Neue Ideen des nächsten Jahres machten Gebrauch vom rekursiven Aufruf der Funktion „main( )", verschachtelten Kommentaren, die keine waren, und wieder vom schweren Einsatz des Preprocessors. So wurden selbst dessen Befehle neu definiert, was allerdings nicht mehr jeder Compiler mitmacht, ebenso wie mancher „lexical analyser" (der Teil des Compilers, der elementare Zeichenfolgen, etwa Operatoren und Schlüsselworte, erkennt) einen ersetzten Text nicht mit angrenzenden Zeichen zu Tokens verschmilzt. Richtig unübersichtlich wird es auch, wenn im mit „#define" vereinbarten Textersatz Klammern oder Kommentarbegrenzer nicht paarweise zueinander passend definiert werden. Einige dieser Ideen finden sich im Beispielprogramm für jenes Jahr wieder. Es beruht unter anderem darauf, daß der Underscore „_" in den Namen von Bezeichnern als gleichberechtigt zu Buchstaben behandelt wird. Rufen Sie das Programm ohne Parameter auf und geben Sie dann eine Textzeile ein. Beenden Sie die Ausführung mit EOF durch CTRL-N. Kommentar der Schiedsrichter: „The program itself looks like tty noise."

1986: Das Programm von Jim Hague gehört zu den wenigen, die durchaus einen praktischen Zweck erfüllen: Es gibt den Text der Standardeingabe als Morsecode wieder aus. Zu den Kunstwerken gehört das andere Beispielprogramm des gleichen Jahres. Sie werden es ein wenig modifizieren müssen, wenn Sie es zum Laufen bringen wollen, da mancher

C-Compiler andernfalls Fehlermeldungen ausgibt. Außerdem wird empfohlen, das Executable nicht auszuführen, bevor Sie nicht jede Zeile verstanden haben. Bereiten Sie sich ansonsten auf den Drei-Finger-Trick vor. Ein weiterer Preisträger von 1986 konnte unter C, Fortran 77 und der Bourne Shell ausgeführt werden und gab jedesmal „Hello, World!" aus!
Ein vollständiges Othello-Programm (Reversi) wurde 1987 innerhalb des im Vorjahr auf 1024 Bytes heraufgesetzten Limits realisiert.

Regeln für geordneten Wildwuchs

Mit dieser Aufstockung ging eine Präzisierung der Regeln einher: So gab es Extrapunkte, wenn das Programm unbeanstandet den C-Scharfrichter „lint" überstand. Auch führt mittlerweile eine simple Überbeanspruchung von „#define" zur Abwertung. Dieses bisher letzte Jahr stellt die Krönung der Verwirrung dar: So schaffte es Mark Biggar durch seinen Beitrag, daß die Regeln wieder einmal geändert wurden — nachdem er einen Preis für seine wahrhaft geniale Idee einheimsen konnte: Sein Programm besteht einfach nur aus ...
P;
... wobei manchen C-Compilern sogar das Semikolon noch zuviel ist. Daß hier noch irgendwo ein Haken sein mußte, war klar. Mark Biggar nutzt die Möglichkeit des Compilers, beim Aufruf noch Symbole zu definieren, was meist dem Debugging und der Anpassung flexiblen Quelltextes an unterschiedliche Systeme dient. Das wirkliche Programm liegt also im Aufruf und lautet ...

```
cc -DC="R>0" -DI="if(T)O"
   - DO="c=write(1,&c,1);"\
-DP="main(){X}" - DR="read
   (0,&c,1)" -DT="c!=015"\
-DW="while(C)I" - DX="
   char c;W" biggar.c
```

Ansonsten filtert sein Programm CRs aus ASCII-Texten, aber darauf kommt es wahrlich nicht mehr an. Durch Änderung der Compilezeile kann dieses Programm in jede Richtung geändert werden. Um wieder die Schiedsrichter zu Wort kommen zu lassen: „We will let Mark get away with this truely sick entry this time, but for the future on we have placed a limit on the size of a compile line". Diesmal wollen sie's also noch durchgehen lassen, in Zukunft gibt's aber keine überlangen Compileraufrufe mehr ...

Das Grauen muß jeden Modula-Fan beim Lesen des Beitrags von Spencer Hines packen. Es dürfte sich wohl um das ultimate Anwendungsbeispiel zur GOTO-Anweisung handeln. Geben Sie dem Programm einmal den Namen seines eigenen Quellfiles als Parameter an. Zum Abschluß noch ein weiteres Kunstwerk, das diesmal aber problemlos zu compilieren ist. Glauben Sie nicht, daß das Programm nur etwa den Inhalt des Feldes „redivider" ausgibt — die Sache ist schon etwas subtiler. Vielleicht sollte AmigaWelt in Zukunft die Lesereinsendungen auf die gleichen Kriterien untersuchen — es finden sich sicherlich hinreichend viele Beiträge, die sich auch unbeabsichtigt dafür qualifizieren, nicht nur im Bereich der Listings.

(Ralph Babel)

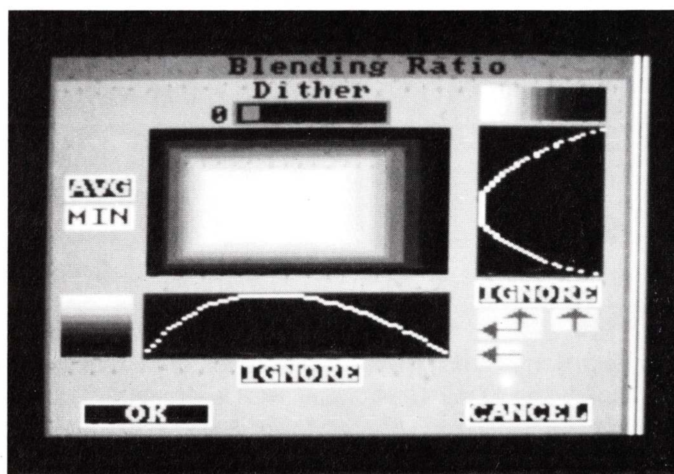
PHOTON PAINT — ein Grafik-HAMmer

Neues von der HAM-Front kommt aus Israel. Ein Programm, das alle bisherigen HAM-Utilities in den Schatten stellt. 4096 Farben, einzigartige Brushfunktionen zusammen mit allen gewohnten Standards bietet die Software, die von BAZB'O SOFT programmiert und von MICRO ILLUSION vertrieben wird.

Schon die Vorankündigungen hatten die Neugierde an Photon Paint geweckt. Um mit PHOTON PAINT arbeiten zu können, sind keine besonderen Vorarbeiten nötig. Das Programm ist nicht kopiergeschützt und somit einfach auf Festplatte zu installieren. Nach dem Start von der Workbench erscheint das schwarze Malbrett und im oberen Teil des Bildschirms ein Menü. Dieses Fast-Menü läßt sich jederzeit am Bildschirm nach oben und unten schieben. Zusätzlich erreicht man über die rechte Maustaste eine Reihe von Pulldown-Menüs, wie man dies von Amiga-Programmen gewohnt ist. In der linken Hälfte des Fast-Menüs finden sich vier Reihen à 16 Farben. Benötigt man weitere Farben, so kann man durch Anklicken der beiden Pfeile rechts oben in der Ecke, die Farbauswahlpalette hervorzaubern. Hier können dann Farben, Farbverläufe und die gewohnten Features, wie Helligkeit und Sättigung eingestellt werden. Anordnung und Symbole des rechten Teils des Menüs sind stark an die Benutzerführung von DPaint angelehnt. PHOTON PAINT verwendet fast überall die gleiche Tastaturbelegung wie DPaint. Für professionelle Anwender, die nicht jedesmal auf das Menü zugreifen wollen um eine Funktion aufzurufen, sehr vorteilhaft.

Overscan bis Interlace: Photon Paint ist ein Allround-Talent

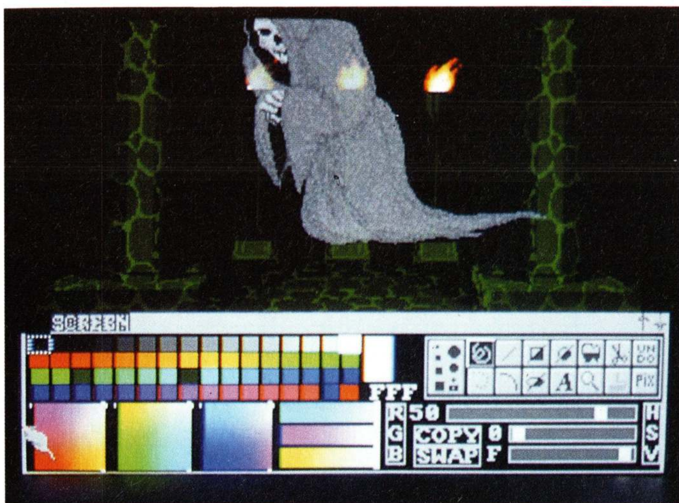
Bei den ersten Malversuchen stellt man fest, daß nur eine NTSC Bildschirm zur Verfügung steht. Ein Blick ins Handbuch und SCREEN MODE aus dem Pulldown PREFS ausgewählt, eröffnet die Möglichkeit, zwischen NTSC und PAL Bildschirm, INTERLACE oder NONINTERLACE-Modus und sogar OVERSCAN zu wählen. Diese und andere Voreinstellungen können dann gesichert werden. Diese Daten werden in einer speziellen Datei gespeichert, deren Aufbau am Ende des Handbuchs, zusammen mit dem Aufbau der IFF-Files beschrieben wird. PHOTON PAINT kann alle IFF-Bilder, egal welcher Auflösung, laden. Diese Bilder werden dabei in das HAM-Format umgewandelt. Das Laden und Abspeichern ist eine der wenigen Schwächen des Programms. Der Requester erscheint — etwas ungewöhnlich — auf eigenem Bildschirm. Bis dann allerdings das Inhaltsverzeichnis aufgebaut ist, vergeht einfach zuviel Zeit. Hier sollten sich die Programmierer noch etwas einfallen lassen.



In diesem Menü werden die Konturenübergänge des Blend-Modus definiert.

PHOTON PAINT stellt alle Zeichen und Malfunktionen zur Verfügung die man als Grafiker benötigt. Ab und zu muß man auf ein Ergebnis etwas warten, was natürlich bei so vielen Farben verständlich ist. Im Vergleich zu anderen HAM-Programmen ist PHOTON PAINT mit Sicherheit das schnellste. Zusätzlich ist zu erwähnen, daß das Programm bei größeren Berechnungen in der Kopfzeile des Menüs einen Countdown ablaufen läßt — für die allzu ungeduldigen User.

Bislang unbekannte Möglichkeiten in einem Malprogramm bietet die Software in puncto Brushes. Anstrahlen mit einer Lichtquelle, ist eine ganz herausragende Brushfunktion. Brush in Würfelform umsetzen? Mit PHOTON PAINT kein Problem. Natürlich sind die



Schatteneffekte lassen sich mit 4096 Farben leicht realisieren.

„normalen“ Optionen, wie drehen, zoomen und flippen vorhanden. Die Brushfunktionen sind das Optimale, was es zur Zeit für den Amiga gibt. Abenteuerlich wird es im BLEND-MODUS. Mit Hilfe dieser Option könne durchscheinende Brushes und Farbverläufe erzeugt werden.

Großes Talent im Jonglieren von Brushes

In PHOTON PAINT gibt es keine SMEAR, SMOOTH und ähnliche Funktionen wie in DPaint. Allerdings wären diese im HAM-Modus auch nur äußert schwierig zu realisieren.

Ein Vergleich von PHOTON PAINT mit anderen HAM-Programmen ist eigentlich nicht möglich, da diese im Verhältnis zu PHOTON PAINT wie aus der Steinzeit wirken. PHOTON PAINT und DPaint sind auch nicht zu vergleichen. Vieles läßt sich in DPaint sehr viel schneller, einfacher und eindrucksvoller malen, als in PHOTON PAINT, da DPaint sich nicht mit dem Speicher- und Rechenaufwendigem HAM-Modus belastet. Die beiden Programme ergänzen sich, ersetzen einander aber nicht. PHOTON PAINT ist für alle diejenigen gedacht, die mit sehr viele Farben arbeiten wollen — zum Beispiel mit digitalisierten Bildern. Die Autorin des Manuals zeigt ihren vielen männlichen Kollegen, wie man ein gutes Handbuch schreibt. Leider nur in Englisch wird der User hervorragend an das Programm herangeführt.

(Robert Wäger)

Info: Computergrafik Lechner, Planegger Straße 1, 8000 München 60, Tel.: 089 / 834 05 91, Preis 249 Mark DTM, Poststr. 25, 6200 Wiesbaden, Tel.: 0 61 21/56 00 84

KOPIER-PROGRAMME IM RING

Ehrlich erworbene Original-Software ist ebensowenig gegen Beschädigungen gefeit, wie selbstangelegte Datendisketten. Fingerabdrücke, Zigarettenasche, Kaffee oder ein Format-Befehl zur falschen Zeit können die Daten in die ewigen Jagdgründe schicken.

Wohl dem, der eine Sicherheitskopie seines wertvollen Originals angelegt hat. Viele Softwarefirmen sind dazu übergegangen, ihre Anwenderprogramme ohne aufwendigen Kopierschutz zu vertreiben. Dies erleichtert die Anfertigung eines dringend notwendigen Backups und senkt die Herstellungskosten. Was macht der Benutzer aber bei Programmen, die sich einfach nicht kopieren lassen? Er hat entweder die Möglichkeit, sich über den schwarzen Markt eine Raubkopie zu besorgen, deren Kopierschutz entfernt wurde (die denkbar schlechteste Form einer „Sicherheitskopie“). Die andere Chance für den geplagten User ist, sich auf die Zusicherung des Herstellers zu verlassen, daß ihm im Falle eines Defektes eine neue Programmdiskette zugesendet werde. Eine Firma, die Software kommerziell einsetzt, kann während der Umtauschzeit Betriebsferien einlegen. Immer mehr Software-Firmen versuchen nun aus diesem Dilemma Kapital zu schlagen und bieten Kopierprogramme an, die angeblich „jeden Kopierschutz überlisten“, ausgenommen den eigenen. Fast jeden Monat erscheint nun ein neues Programm dieser Art auf dem Markt. Amiga-Welt wollte wissen, was die Versprechungen der Hersteller wert sind. Folgende Programme konnten sich für den Test qualifizieren:

A-Copy

Alt, aber gut, so kann man das Public-Domain-Programm „A-Copy“ bezeichnen. Obwohl nicht neu oder besonders schnell, kopiert es einige Orginalprogramme, bei denen professionelle Kopierprogramme größte Schwierigkeiten haben. Ausreichende Möglichkeiten bietet das knappe Auswahlmenü. Die Palette der Optionen reicht von NormalCopy, über Nibble- und ParameterCopy bis zum IBM-Format. Start- und Zielzylinder können frei von 0 bis 81 gewählt werden. Dabei kann aber nur ein Ziellaufwerk angegeben werden.Da dieses Programm frei von Rechten ist (Public Domain), sollte es kein Amiga-User missen.

Amiga-Copy 1.2

Große Überraschung: Data Becker, Vertreiber von zahl- ▶

Programmname	1	2	3	4	5	6	7	8	9	10	11	12	13	
A-Copy	2,28	—	2,28	—	—	—	2,28	—	1	81	ja	nein	ja	
Amiga-Copy	3,13	1,50	DEEPCOPY I 3,18		DEEPCOPY II 3,18				3	81	ja	ja	ja	
Fast Lightning	1,09	1,50	TOPSECRET 2,29		Hulla Baloo 2,26		MATCHPOINT 1,37		3	81	ja	nein	nein	
Marauder II	1,25	2,24	—	—	2,40	3,45	1,25	2,54	3	82	nein	ja	nein	
Project D	1,25	2,14	1,25	2,14	2,00	3,16	—	—	3	82	ja	nein	ja	
Superkit	1,37	2,31	—	—	1,44	1,18	—	—	3	81	ja	nein	nein	
Turbo Copy	1,21	2,08	—	—	—	—	—	—	1	79	nein	nein	nein	
White Lightning	1,55	2,33	—	—	—	—	—	—	1*	79	ja	ja	nein	
Amiga Tools	1,40	—	—	—	—	—	—	—	1	79	nein	nein	nein	Fast Format 0,55 Virus Finder 1,27
Parameter Copier	—	—	1,26	—	—	—	—	—	1	81	ja	nein	ja	

Zeiten in Minuten

* nur internes Laufwerk

Bedeutung:
1) Normalcopy ohne Verify
2) Normalcopy mit Verify
3) Parametercopy ohne Verify
4) Parametercopy mit Verify
5) Indexcopy ohne Verify
6) Indexcopy mit Verify
7) Nibblecopy ohne Verify
8) Nibblecopy mit Verify
9) Wieviel Ziellaufwerke maximal
10) Bis welchen Zylinder maximal
11) Zylinderzahl frei einstellbar
12) Kopieren aus RAM
13) Fremdformate

reicher Originalsoftware, bietet ein eigenes Kopierprogramm an. Auf der beiliegenden Diskette zum neuen Amiga-Schmöker „Das große Floppybuch", befindet sich Amiga-Copy 1.2. Neben einem der schnellsten Kopiermodi (FastCopy) für ungeschützte Disketten, gibt es noch zwei verschiedene DEEP-Copy-Einstellungen. Mit DEEPCOPY I lassen sich sowohl Fremdformate als auch kopiergeschützte Disketten kopieren. Bei DEEPCOPY II wird ein anderes Verfahren zur Auffindung der Track-Lücke verwendet. Leider arbeitet das Programm hierbei nur mit einem Ziellaufwerk. Mit dem Menüpunkt „Synccorrection" kann mit oder ohne Sync-Korrektur kopiert werden. Mit der Funktion „How many tries" kann auf die Häufigkeit der Schreibversuche Einfluß genommen werden. Nicht jede Zieldiskette muß defekt sein, falls beim Schreibvorgang ein Verify-Error ausgegeben wird. Weiterhin besteht, falls nur ein Ziellaufwerk angewählt wurde, die Möglichkeit, Daten auf mehrere Zieldisketten zu schreiben. Ein Teil der Daten wird ins RAM eingelesen, auf einer oder mehreren Zieldisketten abgespeichert, bevor die Quelldiskette wieder angefordert wird. Besitzer einer Speichererweiterung können somit eine Diskette komplett in das RAM einlesen und sofort beliebig viele Kopien anfertigen, ohne die Master durch unnötige Schreibversuche zu traktieren.

Amiga-Tools

Auf der CeBIT präsentierte die Firma CSJ Computersoft GmbH eine kleine Sammlung von Utilities unter dem Namen „Amiga Tools". Die neuere Version enthält vier verschiedene Hilfsmittel für den Amiga-Anwender: RAM DELETER; hier wird nicht nur das FASTRAM komplett gelöscht, sondern es werden auch noch bestimmte Adressen umkopiert, so daß Programme, die mit erweitertem RAM in Verbindung mit dem normalen NoFast-Mem-Befehl nicht lauffähig waren, nun einwandfrei funktionieren.

FAST FORMAT, mit dem neue Disketten in bisher nicht bekannter Geschwindigkeit formatiert werden. Neu ist hier auch die Möglichkeit, mit mehr als einem Laufwerk gleichzeitig zu formatieren. Disketten, die nachweislich einen Hard-Error aufweisen, werden allerdings als ordnungsgemäß formatiert ausgewiesen. Aus diesem Grunde sollten wirklich nur einwandfrei funktionierende Datenträger mit „Fast Format" behandelt werden.

Kleiner Schwächeanfall: falsche Meldung bei mißlungener Formatierung

DISKCOPY — nicht das schnellste und nicht als das stärkste Geschütz für Hard-Protected Programme zu betrachten. Es läßt sich aber ohne weiteres mit auf dem Markt befindlichen „normalen" Kopierprogrammen vergleichen.

FAST LIGHTNING

Mit diesem Programm hat die Firma Versalia das derzeit schnellste Kopierprogramm auf den Markt gebracht. Unter dem Punkt „Supersonic" ohne Verify macht „Fast Lightning" eine Kopie von einer ungeschützten Diskette in ganzen 69 Sekunden. Die anderen Modi haben etwas exotische Namen, wie „Top Secret", „Hulla Baloo" und „Match Point", sind aber mit den normalen Features wie Index und Nibble vergleichbar.

MARAUDER II Brainfile 10

Marauder von Discovery Software ist zwar der Oldie unter den Kopierprogrammen, aber unter der Haube geht die technische Entwicklung ständig weiter. Als eines der ersten wirklich guten Kopierprogramme für den Amiga schon zu Zeiten, als es noch keine Amiga 500 oder 2000 gab, hat sich die Herstellerfirma von „Marauder" nicht auf ihren Lorbeeren ausgeruht, sondern immer wieder eine neue, verbesserte Version auf den Markt gebracht. Inzwischen liegt mit der Brainfile 10-Version die neueste Ausgabe vor. Registrierte Besitzer eines Marauder-Programmes können sich übrigens gegen einen geringen Betrag immer die neueste Version als Update zusenden lassen.

Marauder arbeitet sowohl im Normal-, als auch im Index- oder Nibble-Modus. Weiterhin sind Analytical- oder Verbatim-Modus anwählbar, wobei ersterer die meisten Kopierschutzmechanismen überwindet und zweiterer eine Eins-zu-Eins Kopie der Diskette erstellt. Selbstverständlich sind alle Kopiermodi sowohl mit als auch ohne Verify-Option zu benutzen. Auf jeder Diskette der Brainfile-Ausgaben befindet sich eine komplette Liste von derzeitig kopierbaren Orginalprogrammen, welche mit jeder neuen Ausgabe erweitert wird.

Als „kleine" Beigabe finden sich auf Diskette noch einige Hilfsprogramme wie „Speed Check", „DiskErr", „Disk-Wipe" und „Decoder". Herauszuheben ist hier der Decoder, der von einigen geschützten Programmen entschützte Files anlegt, die dann sogar auf Harddisk installiert werden können. „Marauder II" ist auf Festplatte installierbar. Definiert man das Sourcelaufwerk gleichzeitig als Ziel, wird der Disketteninhalt bei genügendem Speicher ins RAM gelesen und kann ohne neues Auslesen der Quelle beliebig oft kopiert werden. In der momentan vorhandenen Version können das die vier möglichen Laufwerke leider nicht.

Parameter Copier 1.03

Wie der Name schon verrät, handelt es sich hierbei um eine besondere Art eines Kopierprogramms. Anders als bei den vorher besprochenen Programmen, hat der Anwender nun die Möglichkeit, sämtliche diskettenspezifischen Parameter selbst einzustellen. Für jede Spur können in einem Menü acht Parameter separat eingestellt werden:

Eigener Kopierschutz überlistet die Fähigkeiten des Programms

SYNC-Word; hier wird festgelegt, bei welcher DMA-Übertragung begonnen werden soll. Es ist aber auch möglich, ohne SYNC zu kopieren. Da besonders die neueren Kopierschutzverfahren auf veränderten SYNCS beruhen, ist diesem Parameterwert große Bedeutung zuzumessen.

INDEX-Pulse; falls eine wirkliche eins-zu-eins-Kopie entstehen soll, so muß ohne SYNC-Word (NSNC), dafür aber mit Index gearbeitet werden (z.B. Parallel-Format).

LENGTH; hier wird die Länge des zu kopierenden Tracks angegeben (in Worten, also Länge in Bytes/2).

PRECOMPENSATION; da der Amiga auch ohne DOS auf die Diskette zugreifen kann, gibt es zwei verschiedene Formate, Daten auf Diskette abzuspeichern. Das GCR-Format (Group Code Recording) und das MFM-Format (Pulse Code Modulation). Zum Verschlüsseln der Daten verwendet der Amiga die MFM-Codierung.

TIME; hier ist die Länge (0, 120ms, 280ms oder 560ms) der Precompensation anzugeben.

SYNC ON MSB; dieser Wert ist nur bei Formaten zu setzen, bei denen jedes Datenbyte ein gesetztes siebtes Bit (MSB = most significant bit) haben muß; siehe Apple-GCR.

SPEED; dies ist die Zeit, die der Controller für ein Bit verwendet. Es besteht die Wahl zwischen 4ms, vor allem bei GCR verwendet und 2ms als Standard-Wert.

FORMAT, der Amiga kennt zwei Formate, A-DOS = Amiga-DOS und N-DOS = No-DOS. A-DOS ist die vom Amiga normalerweise verwendete Einstellung, da aber auch Fremdformate verarbeitet werden können, ist die N-DOS Option sehr wichtig. All diese Funktionen zeigen, daß es sich hier um kein Kopierprogramm im eigentlichen Sinne handelt, sondern um ein ausgereiftes Tool, um von kopiergeschützten Orginalen Sicherheitskopien für den eigenen (!) Gebrauch anzufertigen. Als kleine Beigabe enthält die Programmdiskette bereits einige fertige Parametereinstellungen, die durch einfaches Laden aufgerufen werden können. Ebenso können selbsterstellte Parameter auf Diskette gespeichert werden. Damit für den Kopiervorgang die richtigen Parameter eingestellt werden können, sind weitere Hilfsprogramme, wie Diskscanner, Errorchekker und Speedometer auf der Programmdiskette enthalten. Parameter Copier V1.03 hilft aber auch dem fortgeschrittenen Anfänger, um tiefer in die Geheimnisse der Diskette einzusteigen. Es versteht sich von selbst, daß die Programmdiskette kopiergeschützt ist und nicht mit sich selbst kopiert werden kann. Es sei denn, jemand schafft es, die Parameter auszulesen. Beachtet man auch den für dieses Programm sehr niedrigen Preis von DM 18.—, zzgl. Porto und Verpackung, so dürfte die Anschaffung keine Frage mehr sein.

PROJECT D V 1.0C

Mit diesem Programm bietet Ben Fuller eine konsequente Weiterführung von „Super-

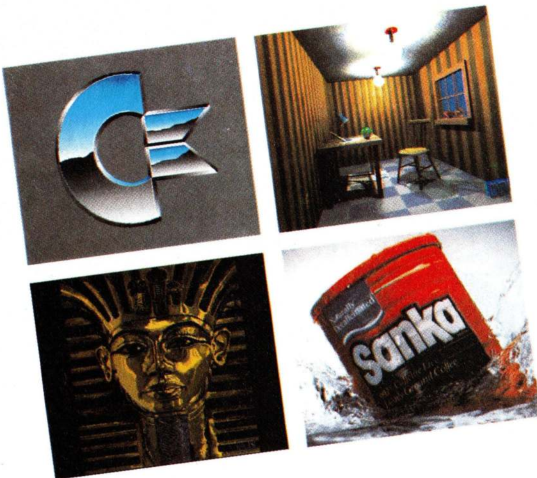
Fortsetzung auf Seite 75 ▶

Desktop Publishing auf dem Weg zur Ernsthaftigkeit:

Professional Page und Shakespeare im Vergleich

Zwei Neuerscheinungen wollen dem Amiga in puncto Desktop Publishing nicht nur Sprüche sondern Druckreifes entlocken: Shakespeare von Infinity Software und Professional Page von Gold Disk. Ob sich das Warten gelohnt hat, zeigt der Vergleichstest.

DEN GEDANKEN IHRE FREIHEIT
DER KREATIVITÄT IHR WERKZEUG





Direkt von der Diskette in die Satzmaschine: Commodores IFABO-Prospekt enthält Farbgrafiken, die ohne Reproarbeiten, nur mit Professional Page realisiert wurden.

Professional Page ist der stark überarbeitete und erweiterte große Bruder von Pagesetter, Shakespeare eine komplette Neuerscheinung. Beide Programme erheben den Anspruch, für professionelles Arbeiten tauglich zu sein. Ohne störenden Kopierschutz (die Festplattenbesitzer werden es danken), mit englischem Handbuch, werden beide Programme seit kurzem in Deutschland ausgeliefert. Professional Page enthält sogar ein Batchfile zum Installieren der benötigten Dateien auf Festplatte.

Anspruch eines DTP-Programms ist, die vielfältigen Arbeitsgänge vom Manuskript bis zur fertigen Druckvorlage zusammenzufassen. Texte werden hierbei nur einmal erfaßt, können dann korrigiert und in Layout-Seiten integriert werden. Vom Benutzer verlangt das Kenntnisse über Seitenaufbau und -gestaltung und vom Programm weitestgehende Unterstützung der Wünsche des Benutzers. Der letzte Arbeitsgang, Erzeugung der druckfähigen Vorlage, läßt sich am saubersten mit einem postscriptfähigen Laserdrucker realisieren. Postscript ist eine Seitenbeschreibungssprache, die weite Verbreitung im Druckgewerbe gefunden hat. Viele Fotosatzmaschinen verstehen sie, zum Beispiel die Linotype, und selbstverständlich die Mehrzahl der Laserdrucker. Diese Geräte besitzen eigene Mikroprozessoren, mit denen sie Datenströme, die in Postscript organisiert sind, in hochauflösende Bitmaps umsetzen und dann ausdrucken.
Professional Page bedient

sich ausschließlich dieser zum Standard avancierenden Ausgabeform. Shakespeare unterstützt auch andere Drucker mit den Druckertreibern der Workbench 1.3. Diese werden auf der Programmdiskette von Shakespeare inclusive einer nicht der aktuellen Gammaversion entsprechenden Ausführung der Preferences 1.3 mitgeliefert. Die neuen Druckertreiber sind schneller als die der Version 1.2 und in der Ausgabequalität ein wenig hochwertiger, vor allem wenn man die neu angefügte Druckoption Antialiasing nutzt. Antialiasing nutzt eine Klasse von Algorithmen, die dazu dienen, die Nachteile einer relativ geringen Auflösung wegzutricksen. Shakespeare druckt die Vorlagen als normalen Grafikausdruck der Workbench aus. Intern hält der Computer aber bestenfalls eine Auflösung von 640 mal 512 Punkten parat.

Dagegen liegt die maximal mögliche Auflösung eines 24-Nadel-Druckers bei über 2500 mal 3900 Punkten auf einer DIN A4 Seite. Ein Amigapixel wird daher durch mehrere Druckerpins präsentiert, was zu deutlichen und häßlichen Treppchenstrukturen führt. Antialiasing bügelt die holprige Bildschirmauflösung glatt. Treppchen, die bei diagonalen Linien entstehen, werden geglättet. Noch immer keine optimale Lösung für ein Publikationsprogramm, aber ein Gewinn für jeden, der sich keinen Postscriptdrucker zulegen will.

Professional Page ist in seiner Auflösung auf den Interlace-Modus festgelegt. Shakespeare läßt dem Anwender die Wahl der Qual: Im Noninterlaced-Modus werden die Proportionen des Dokuments auf dem Bildschirm verzerrt dargestellt. Unerläßlich für die Arbeit mit beiden Programmen, ist daher eine Hardware-Erweiterung, die den flimmernden Interlace-Bildschirm zur Ruhe bringt. Ein Bericht über diesen Flicker-Fixer befindet sich in dieser Ausgabe.
Die Konfiguration für ein optimales Arbeiten sollte also ein Amiga 2000 mit FlickerFixer, ein multisynchroner Monitor, zweites Laufwerk oder Festplatte und ein postscriptfähiger Laserdrucker sein. Als Richtschnur für eine Einordnung der Leistungsfähigkeit der beiden DTP-Programme diente der Ventura Publisher auf einem PC/AT Rechner mit Harddisk und viel RAM, das als große RAM-Disk genutzt, wichtig für die Verarbeitungsgeschwindigkeit der Venturabefehle ist.

Piktogramme repräsentieren die Arbeitsmittel des Bildschirmlayouters

Nach dem Laden von Professional Page erscheint auf dem Bildschirm ein leerer Arbeitsplatz mit den Piktogrammen für einige Layoutwerkzeuge. Erster Arbeitsschritt ist die Definition einer neuen Arbeitsseite.
Dies geschieht durch die Anwahl des Menüpunktes Page/Create/from default. Es erscheint ein Requester, in dem das Format der neuen Seite festgelegt werden kann: die Breite der Ränder, Anzahl der Spalten und die Breite der Lücken zwischen ihnen.
Vorher sollten nach Bedarf in dem Menüpunkt Preferences/Layout Tools die Einheit der Bemaßung, die Benutzung eines Gitters und des Lineals, und die Darstellung von Hilfslinien zur Seiteneinteilung voreingestellt werden.
Zum groben Bearbeiten der Seite und für Feinarbeiten lassen sich jeweils verschiedene Vergrößerungsstufen wählen. Bei Vergrößerungen ab 100 % lassen sich Details und Schrift klar erkennen, die geringeren Vergrößerungen dienen dazu, die Gesamtgestaltung der Seite zu kontrollieren. Bei den niedrigen Vergrößerungen wandert man mit den Cursortasten auf der Seite umher, oder man zieht mit der Maus das Seitenausschnittgadget auf der die gesamte Seite repräsentierenden Fläche in der Werkzeugleiste umher.
Wenn jetzt Text und Grafik auf dieser neuen Seite dargestellt werden sollen, müssen sogenannte Boxen eingerichtet werden. Sie dienen als Behälter für Texte, Zeichnungen oder Bilder. Dazu aktiviert man aus der Reihe der Werkzeugpiktogramme das Boxenwerkzeug und beginnt Boxen auf die Seite zu legen.
Voreingestellte, die Seite aufteilende Hilfslinien sind für den Platz, den eine Box real auf der Seite einnimmt, nicht verbindlich, sie dienen lediglich als Richtschnur für die optische Gliederung. Die Option zum Einrasten auf das Gitter hilft beim präzisen Plazieren der Boxen ungemein. Boxen lassen sich in Größe und Form an Size Gadgets am Rand der Box oder im Menupunkt Box/Alter verändern, wobei letzterer ein maßstabsgerechtes Zoomen erlaubt.
Es ist kein Problem, überflüssige Boxen wieder zu löschen oder eine Kopie von bestehenden Boxen anzufertigen. Boxen, die in einem gewissen Zusammenhang stehen, lassen sich zu einer logischen Einheit zusammenzufassen und gemeinsam manipulieren. Man kann auf der Oberfläche von Professional Page innerhalb der Boxen Texte mit einem integrierten Texteditor editieren, der sich durch Anklicken des Textwerkzeugs aktivieren läßt, und in der ausgewählten Box aktiv wird. Dieser Texteditor kann nur minimalste Ansprüche in Hinblick auf die Tippgeschwindigkeit befriedigen. Er ist zum flüssigen Bearbeiten von ganzen Dokumenten völlig ungeeignet; für Bildunterschriften, Titel und Korrekturen ausreichend.
Zum Glück unterstützt Professional Page das Laden von Texten fremder Textprogramme und kann sogar deren Formatanweisungen interpretieren. So werden WordPerfect, Scribble!, TextCraft, TextCraft+ und ASCII-Text unterstützt. Nachdem im Menüpunkt Preferences/Text das Textformat eingestellt ist, kann man mit der Funktion Projekt/Import/Text Textdateien in Professional Page laden.
Die Texte sind nun im Paste Zwischenspeicher eingeladen. Jetzt kann man eine der Boxen auf der Seite anwählen und nach Wahl der Schrifttypen und des Erscheinungsbildes des Textes in die Box hineinlaufen lassen. Sollte der Text größer sein als die Box, so kann man weitere Boxen an diese erste Textbox anhängen, der Text fließt automatisch in diese angehängten Boxen nach. Automatische Silbentrennung jedoch wird im Moment leider nur nach den amerikanischen Trennungsregeln unterstützt. Hier muß der User nachkorrigieren. Eine Seiteneditierung ohne ▶



dieses Feature wäre wie Autofahren ohne Reifen. Texte und Grafiken bei neuen Entwurfsideen interaktiv umzuformatieren, wird mit Professional Page zum Kinderspiel. Alle Funktionen eines Programms wie Ventura Publisher werden von Professional Page zur Verfügung gestellt, Optionen wie Kerning und Einzüge gehören zum Standardrepertoir. Mit dem integrierten Zeichenwerkzeug kann der Text mit vektororientierten Grafiken illustriert werden. Dieser Editor bietet die wichtigsten Zeichenfunktionen an, unterstützt verschiedene Flächen- und Linienfüllmuster, so daß man normale Aufgaben schnell erledigen kann.

Aufwendigeres Werkzeug findet man in Aegis-Draw+, dessen Dateien Professional Page versteht. Ob das neue XCad ebenfalls eingebunden wird, ist offen. Hierüber wurde auf der Developers Conference im Mai in Washington verhandelt (dieser Artikel entstand früher). Als dritte Option bietet Professional Page das Einladen von Interleaved Bitmaps im Interleaved File Format (IFF) an. Jedes Amiga-Malprogramm kann daher zur grafischen Vorbereitung des Seitenlayouts genutzt werden. Auch hier kann mit Hilfe der Boxen das Bild in der Größe und Form nachbearbeitet werden.

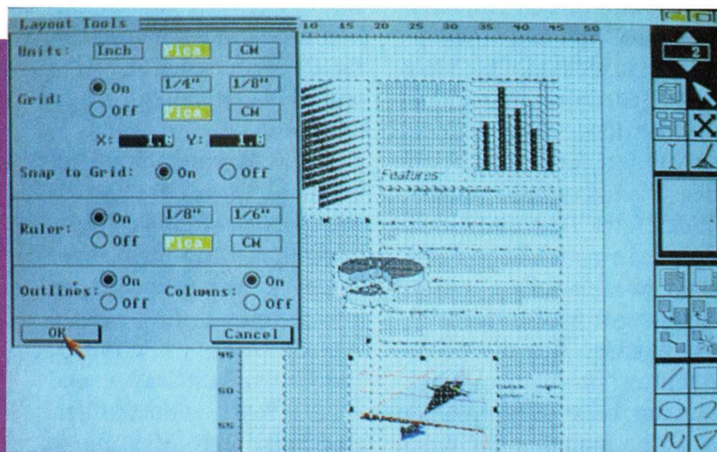
In der aktuellen Version 1.0 kann Profesional Page die Farboption anbieten (Einfügen von HAM-ILBM's!). Erst in der nächsten Version 1.1 (auf der CeBIT '88 schon zu sehen gewesen) wird der im Handbuch schon dokumentierte Colorseperator angeboten, mit dem dann auch vierfarbige Druckvorlagen mit Linotronics möglich sind. Beim Arbeiten mit Professional Page fiel auf, daß der Bildschirmaufbau sehr langsam geschieht, im Vergleich dazu war Ventura Publisher bei fast allen Funktionen einen Tick schneller (wohlgemerkt: ich benutzte eine 4-MB-RAM-Disk, auf die unter anderem das Fontdirectory kopiert war). Das stimmt nachdenklich, zumal der PC/AT keinen Blitter für Grafikoperationen besitzt. Dagegen sind die Gestaltungsmöglichkeiten und die Leistungen bei Professional Page höher zu bewerten. Den Eindruck unterstützt außerdem der mitgelieferte Fonteditor, mit dem man die metrischen Postscriptfonts in der Größe an jeden Bedarf anpassen kann.

Handbücher ersetzten nicht Grundwissen über Layout-Prinzipien

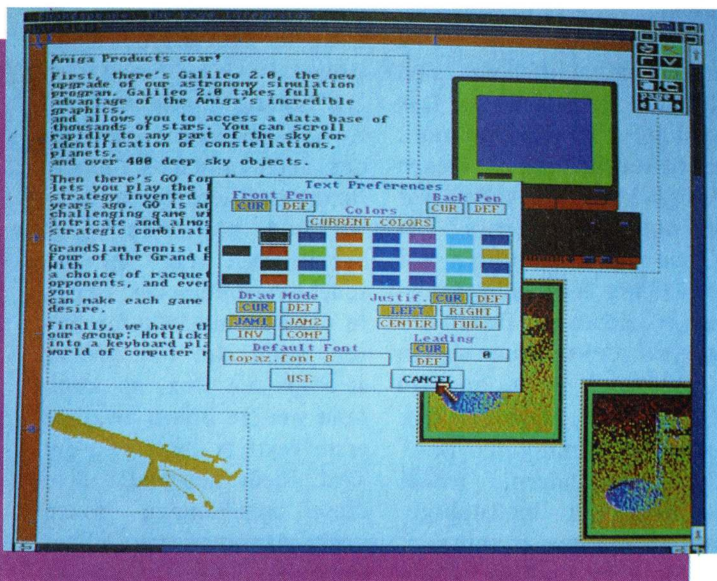
Das Handbuch ist sehr übersichtlich aufgebaut, klar gegliedert und bietet mit seinen hervorragenden Einführungskapiteln einen guten Einstieg in Professional Page. Eine zu knapp geratene Einführung in Layoutprinzipien sollte auf jeden Fall für den völligen DTP-Neuling durch Fachliteratur ergänzt werden.

Viele Farben und buntes Umsortieren der einzelnen Elemente des Seitenlayouts charakterisieren das Seitenlayout einer Seite mit Shakespeare.

Wie schon erwähnt, läuft



Professional Page bietet einen übersichtlichen Arbeitsplatz.



Shakespeare betont das Verspielte und Bunte.

Shakespeare in jedem Auflösungsmodus des Amiga, von 320 mal 256 bis 640 mal 512 interlaced; die Bildschirmdarstellung der Grafikseite ist jedoch für das Arbeiten in der höchsten Auflösung wie in Professional Page ausgelegt, und führt beim Arbeiten mit Shakespeare in niedrigeren Auflösungen zur verzerrten Darstellung der Seite. Die Auflösung läßt sich im Menüpunkt * /Change Resolution einstellen. Die Fähigkeiten des Grafikchips voll ausnutzend, lädt Shakespeare von Anfang an zum farbigen Layout ein.

Stark gewöhnungsbedürftig ist für Amigabenutzer die Benennung und Wirkung der Befehle in der Menüleiste, alles funktioniert etwas anders als gewohnt. Zum Beispiel ist der Requester zum Laden von Daten bei Shakespeare in zwei Fenster geteilt. Im oberen der beiden Fenster greift man auf den Massenspeicher zu, um Daten in einen Puffer zu laden, der mit Hilfe des unteren Fensters verwaltet wird. Der Benutzer greift beim Einfügen von Daten in das Dokument nie direkt auf Disketten zu. Dieses Konzept des sicher gepufferten I/O ist eine Besonderheit, die Programmierer für künftige Projekte in die Überlegung einbeziehen können. Im Gegensatz zu Professional Page beginnt man bei Shakespeare nicht auf einem leeren Arbeitsplatz, sondern auf dem Bildschirm erscheint gleich die erste Seite des Dokuments. Die Voreinstellungen lassen sich jedoch mit Hilfe der Text-Preferences eigenen Vorstellungen anpassen.

Auch bei Shakespeare sollte der Text mit einem Textverarbeitungssystem vorbereitet werden, und nur kleinere Änderungen mit dem WYSIWIG-Texteditor vorgenommen werden. Der Texteditor hat jedoch die Schwäche, den Bildschirmausschnitt der Seite nicht sauber mitzuführen. Die lockere Art, mit der sich Bitmap-Grafiken mit Shakespeare in das Layout einpassen lassen, zeigen die große Stärke des Programms. Das Fehlen eines Gitters zum genauen Plazieren von Gestaltungsteilen, ebenso wie eine nicht vorhandene Lupen- und Zoomfunktion zur Beurteilung der Gesamtseite und das Fehlen eines vektororientierten Grafikwerkzeuges sind klare Mängel.

Einen einfach gangbaren Weg zum Kopieren von Frames (dem Äquivalent zu den Boxen in Professional Page) sucht man genauso vergeblich wie eine Funktion zum Klonieren von Textframes. Mit Copy Frame werden nämlich alle Attribute bis auf die Größe des Frames mitkopiert.

Die erfrischende Unkompliziertheit macht neugierig auf Seitenlayout mit dem eigenen Computer. Beim näheren Hinsehen wird man allerdings enttäuscht, da leistungsfähige Befehle für komplexe Aufgaben fehlen. Auf der anderen Seite ist Shakespeare das erste Programm, bei dem sich die Programmierer Mühe gegeben haben, dem Besitzer von Nadeldruckern das Leben etwas leichter zu machen. Endlich werden geeignete Fonts zu Shakespeare mitgeliefert, die gerade bei Überschriften eine brauchbare Auflösung und ein vernünftiges Ergebnis liefern. Ein schwerwiegender Nachteil ist, daß die Druckgeschwindigkeit von Shakespeare, obwohl es die neuen, schnelleren Druckertreiber der Workbench 1.3 benutzt, so ziemlich die langsamste Druckgeschwindigkeit im Vergleich zu allen bisherigen Amiga Publikationsprogrammen hat.

Lästig: Im Detail steckt der Bug

Leider müssen einige Bemerkungen in Hinsicht auf die Absturzsicherheit von Shakespeare gemacht werden. Der Anwender sollte gerade in der Lernphase, aber auch in der täglichen Benutzung eines Programmes, die Möglichkeit haben, mit dem Programm zu experimentieren. Fehlbedienungen jeder Art sollte ein Programm klaglos wegstecken. Shakespeare kann dieser Forderung noch nicht entsprechen und muß vom Programmierer überarbeitet werden. Zu oft stürzt das Programm infolge einer Fehlbedienung ab.

Professional Page ist in dieser Hinsicht sicherer. Das Lesen des Handbuchs ist bei Shakespeare absolut notwendig, da es eine ganze Menge nicht selbstdokumentierender Funktionen besitzt. Dieses Handbuch ist sehr liebevoll und aufwendig gestaltet und hilft dem Anwender mit drei Lernkapiteln, die schwersten Klippen zu umschiffen. In der rein faktischen Vermittlung steht ihm das Handbuch von Professional Page nicht nach. Im Vergleich zu Ventura Publisher oder der reinen Textverarbeitung WordPerfect erscheint der Umfang beider Handbücher etwas dünn.

Shakespeare erschien als interessante Neuerscheinung auf dem DTP-Markt, ist jedoch zu spezialisiert auf Farbverarbeitung bei durchschnittlichen Anforderungen. Außerdem legt es zu oft Meditationsübungen ein. Wer ein Programm für professionelle Anwendung sucht, sollte auch die Anschaffung passender Hardware nicht scheuen. Professional Page könnte dann das richtige Programm sein. Als preisgünstige Lösung kann man Shakespeare in Betracht ziehen.

(Wilfried Häring)

Info:
Professional Page
Preis: 648 Mark,
Vertrieb: DTM, 6200 Wiesbaden, Poststr. 25, Tel.: 0 61 21/56 00 84

Computergrafik Lechner, 8000 München 60, Planeggerstr. 1, Tel.: 0 89/8 34 05 91

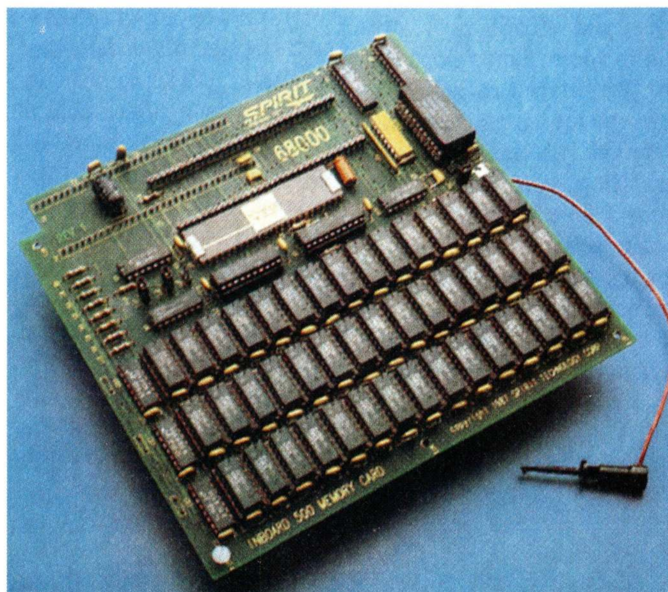
Shakespeare
Preis: 345 Mark,
Vertrieb: Antlantis, Dunantstr. 53, 5030 Hürth 8, Tel.: 0 22 33/4 10 81

Speichererweiterung: Innenboard und Außenboard

Mit jeder neuen Software-Entwicklung wächst der Speicherplatzbedarf. Besonders für die 500er-Besitzer bedeutet das sehr schnell: Eine Erweiterung muß her. Zwei Varianten werden verglichen.



▲ **Einfach zu handhaben, jedoch ein dicker Klotz auf dem Arbeitstisch: Byte Box**



Nach mühsamer Installation verschwindet Inboard 500 im Gehäuse

Von Byte by Byte kommt die Byte Box. Spirit Technology liefert das Inboard 500. Beide erweitern den Speicherbereich des Amiga 500. Hier allerdings enden die Gemeinsamkeiten. Während die Byte Box einfach in den Expansionslot gesteckt wird, muß Inboard 500 in den Sockel des 68000 gesteckt werden. Wir probierten beides aus, um Unterschiede festzustellen.

Die Byte Box präsentiert sich als ein flacher Metallkasten, 3,7 cm hoch, 15 cm breit und 24 cm tief. Sie kann mit 512 KB, einem oder zwei Megabytes oder keinem einzigen Byte bestückt erworben werden. Sie enthält eine eigene Stromversorgung und belastet nicht den Trafo des 500er.

Schonkost für den 500er: Byte Box versorgt sich selbst mit Strom

Der zusätzliche Speicher ist Fastmem, kann daher nicht von den Custom-Chips der neuen Amiga-Generation angesprochen werden. Die Byte Box ist autokonfigurierend — der zusätzliche Speicher befindet sich in dem Bereich, der standardmäßig eingebunden wird und beim Hochfahren des Systems automatisch erkannt wird.

Beim Auspacken findet man nicht nur die Byte Box, sondern noch ein zehnseitiges Handbuch, das ausführlich den Anschluß der Box und die Installation der RAM-Chips erklärt, ferner eine

Diskette mit Diagnosesoftware, um den Speicherbereich zu testen. Einen durchgeschleiften Expansionport wird man vergeblich suchen — auf weitere Geräte an dieser Schnittstelle muß verzichtet werden, falls diese ebenfalls dieses Manko aufweisen.
Inboard 500 besteht aus einer Platine, die unbestückt, mit 512 KB, einem oder 1,5 MByte auf jeden Fall mit Batterie-gepufferter Uhr bestellt werden kann. Es bezieht seine Spannung vom Amiga. Bei Langzeitbetrieb kann das zu Problemen führen, da die interne Stromversorgung begrenzt ist. Obwohl das Board die Logik enthält, damit der Computer den neuen Speicherbereich erkennt, ist es nicht autokonfigurierend. Ursache: Man kann den Speicher bei C00000 oder C80000 anfügen, jedoch nicht bei dem selbst konfigurierenden Bereich, der bei 200000 beginnt. Da das Manual sich auf Diskette befindet, bedarf es noch eines Druckers um eine Hardcopy zu erhalten, damit man während der komplizierten Installation über eine Anleitung verfügt. Während derartiger Innenarbeiten sollte der Computer grundsätzlich ausgeschaltet sein. Wie kann der User dann seine Diskette einlesen?

Am Bus oder im Gehäuse?

Ohne Probleme ließ sich die Byte Box anschließen — in weniger als einer Minute war der aufgemotzte 500er einsatzbereit. Inboard 500 bereitete mehr Mühe beim Einbau. Mit einem Schraubenzieher mußten Schrauben entfernt werden um das Gehäuse zu öffnen, sowie Tastaturanschluß und die Abschirmung abgenommen werden. Danach wurde der 68000-Chip sorgfältig aus dem Sockel gehebelt und in den vorgesehenen Platz im Inboard eingesteckt. Die richtigen Pins zu erkennen ist allerdings nur möglich, wenn man die Platine, die das monochrome Videosignal erzeugt, etwas zur Seite biegt. Jetzt verlangt das Diskettenmanual, daß die drei Kondensatoren C307, C814 und C815 wie auch die RP104-Platine zur Seite gebogen werden. Tief Luft geholt — alle Bauteile überlebten den rabiaten Eingriff. In den leeren Sockel kann nun das Inboard eingesetzt werden. Einen besonders festen Eindruck machte diese Verbindung allerdings nicht. Zwei Plastikschrauben, die als Abstandhalter zwischen Motherboard und Inboard dienen, ruhten statt auf der Platine auf Chips und anderen Bausteinen. Erst als diese Schrauben entfernt wurden, ließ sich das Board vernünftig einstecken. Eine Clip-Verbindung zum Gary-Chip noch, Speichertest und Einstellen der Speicherbereiche mittels Jumper. Endlich war die Prozedur beendet. Dauer der kompletten Installation: nahezu drei Stunden.
Einmal installiert, arbeiteten beide Produkte einwandfrei. Lediglich bei der Inboard-Version mußte einmal die Verbindung zum Gary-Chip erneuert werden. Während der einwöchigen Testdauer mit dem Inboard, konnten keine Probleme mit der Stromversorgung festgestellt werden. Auch eine Überhitzung des Netzgerätes war nicht wahrzunehmen. Dennoch sind Vorbehalte angebracht, denn derartige Probleme treten nicht in der ersten Betriebswoche auf.

Eins plus Eins ist nicht unbedingt zwei

Eine stärkere Stromversorgung ist zu empfehlen. Zum Abschluß wurden beide Erweiterungen zusammen mit der A501-Platine getestet. Mit der Byte Box erreicht der 500er satte 2,5 MBytes (1,5 MByte in der Box und 1 MByte intern). Mit dem A501-Board kooperiert das Inboard nur in der 512-KB-Variante einwandfrei. Das Ein-Megabyte-Inboard hätte den 500er eigentlich auf 1,9 MByte tunen müssen, tatsächlich waren nur 1,7 MByte gemeldet. Nach Auskunft von Spirit Technologies wird dieser Speicherplatz von einigen PAL-Chips auf dem Inboard „gefressen“. Um diesen Verlust zu vermeiden, kann man den Jumper so umstellen, daß der 500er den Zusatzspeicher beim Starten nicht registriert. Mit dem Befehl Addmem lassen sich dann die Speicher wieder zusammenfügen. Das Problem, nicht genau zu wissen, wohin man den Speicher adressieren soll, bleibt.
Die Byte Box dagegen erweitert den 500er genauso, wie man es erwartet. Einfach einstecken und vergessen. Man benötigt nicht einmal ein Manual, das dennoch exzellent ist. Neben der problemreichen Installation und den Schwierigkeiten mit der Stromversorgung kann das Inboard auch positive Seiten aufweisen: Der Expansionport bleibt frei und man benötigt keinen zusätzlichen Platz auf seinem Arbeitstisch. Letztendlich hat man aber mehr Ärger als Vorteile.
Für beide Erweiterungen gilt die Empfehlung, sie unbestückt zu erwerben. Einzeln gekaufte RAM-Chips sind um einiges billiger als die Differenz zur bestückten Version. Deutsche User sind da besser dran als die amerikanischen Amiga-Besitzer: der Distributor verkauft grundsätzlich die unbestückten Versionen und rüstet sie nach Wunsch zum Tagespreis der D-RAMs auf.

(Bob Ryan/deutsch: uk)

Info: Intelligent Memory, Dorsigallee 18, 6000 Frankfurt/M., Telefon: 0 69/41 00 71

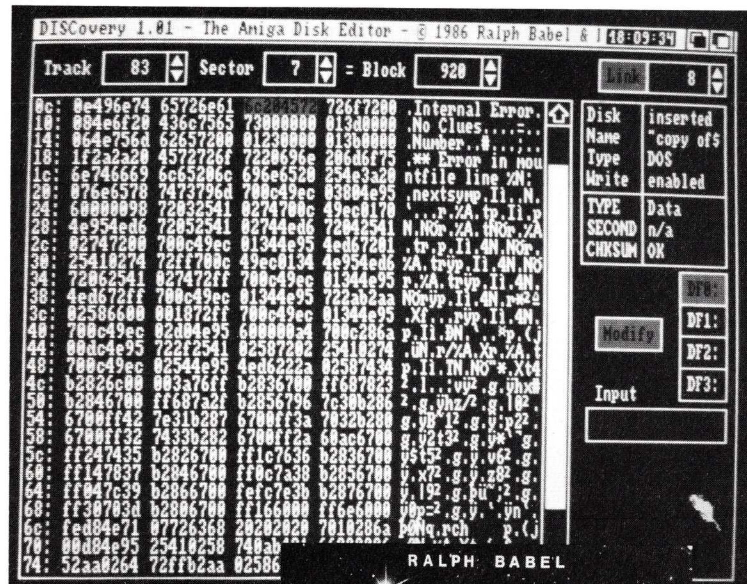
Byte Box
Preis: 698 Mark unbestückt, inklusive Netzteil

Inboard 500
Preis: 498 Mark unbestückt. Interne Uhr für Betrieb ohne A501: 59 Mark

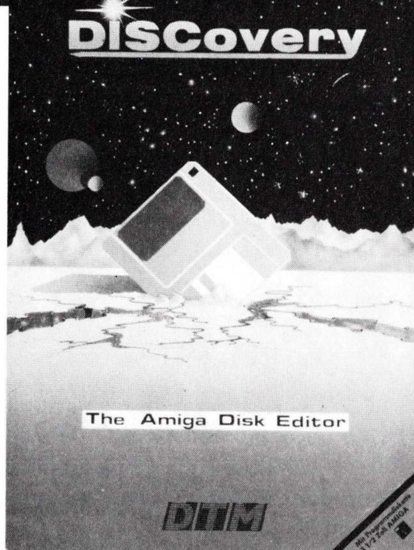
D-RAMs
Typ: 256 KByte × 1, 120 oder besser 150 Nanosekunden, ausgelieferte Version: 41256, Preis 14 bis 18 Mark (Preise stark schwankend)

Mit DISCovery die Diskette durchschaut

Nach knapp einjähriger Arbeit präsentiert Ralph Babel die endgültige Version des Diskettenmonitors „DISCovery". Das Warten hat sich gelohnt — DISCovery besticht mit seinen Leistungen. Mit schlagkräftigen Modulen und einem ausführlichen Handbuch lassen sich nun die Geheimnisse der magnetischen Spuren lüften.







DISCovery bietet umfangreiche Features und ein umfassendes Handbuch.

Das 163seitige Handbuch fällt auf. Ausführlicher kann man Anwendersoftware nicht erläutern. Ganz nebenbei werden die Geheimnisse der beschriebenen Diskette gelüftet.
Blocktypen und die Funktionsweise des Trackdisk-Device werden beschrieben, die MFM-Kodierung (Modified Frequency Modulation) untersucht und die verschiedenen Error-Meldungen erklärt. Das Handbuch alleine könnte schon unter dem Titel „Floppybuch" erscheinen. „DISCovery" soll es ermöglichen, gelöschte und zerstörte Daten wieder zugänglich zu machen. Dazu wird ein breites Angebot an Funktionen geboten. Nach dem Start von „DISCovery" befindet man sich im Hauptmenü und durch einfaches Anklicken von Symbolen können die verschiedenen Funktionsmodule angewählt werden:
Directory-Modul: Nach kurzem Laden wird das Inhaltsverzeichnis der Diskette (DF0 bis DF3 möglich) angezeigt. Hier läßt sich schnell herausfinden, welche Dateien und Unterverzeichnisse zu welchen Verzeichnissen gehören. Jeder Eintrag des Inhaltsverzeichnisses wird mit wichtigen zu ihm gehörenden Attributen angezeigt, die geändert und gelöscht werden können. Desweiteren ist es möglich, Verzeichnisse und Dateien zu schaffen, zu ändern und zu löschen. Es bereitet keine Schwierigkeiten mit „DISCovery" Dateien zu rekonstruieren, doch kann der CLI-Befehl „Diskdoctor" dies bei einfachen Schäden bequemer realisieren, da alles automtisch abläuft. Sollte jedoch aus einer Vielzahl gelöschter Dateien eine bestimmte wiederhergestellt werden, so ist dies nur mit „DISCovery" möglich, da der „Diskdoctor" diese Option nicht beinhaltet.

Reparaturwerkstatt für Sektoren

Sector-Modul: Dieses Modul enthält die interessanteste Befehlssammlung von „DISCovery". Die Dateien und Verzeichnisse können direkt untersucht werden und auftretende Fehler repa-

riert werden. Desweiteren ist es möglich, die einzelnen Datenbytes zu untersuchen und zu ändern. Im großen Anzeigefenster stehen die Daten des angewählten Sektors, dessen Position auf der Diskette über dem Anzeigefenster abzulesen ist. Eine Tabelle zeigt die Adresse der Daten innerhalb des Blockes an. Rechts davon befinden sich die jeweiligen Daten, die in mehreren Spalten angeordnet sind. Daneben steht die Tabelle der ASCII-Äquivalente dieser Daten, dessen Anzeigeformat variiert werden kann. Das Zahlensystem kann geändert werden (octal, decimal oder auch sedecimal), die ASCII-Tabelle und die Adressentabelle lassen sich abschalten und die Zahlen können mit oder ohne Vorzeichen dargestellt werden.

Ausschnitte mit Maus und Tasten wählbar

Der sichtbare Ausschnitt der Anzeige kann mittels der Maus, des Pfeil-Gadgets, des Balkens oder durch die Cursor-Tasten verschoben werden. Eine Besonderheit spielen die Funktionstasten, denn jeder F-Taste kann eine beliebige Sektornummer zugewiesen werden, so daß man mit der Funktionstaste jederzeit zum gemerkten Block zurückspringen kann. Mit Hilfe des Link-Gadgets werden die Blöcke einer Datei verkettet, so daß die Lese-Reihenfolge der wirklichen Block-Reihenfolge in der Datei entspricht. Das Link- Gadget wird auch verwendet um herauszufinden, zu welcher Datei ein beliebiger Block auf der Diskette gehört. Das Input-Gadget dient dazu, den Inhalt eines Blockes zu ändern. Dabei können Text- oder Zahlenfolgen eingegeben werden, wobei die Texteingaben in Anführungsstrichen stehen müssen und bei numerischen Eingaben wieder freie Wahl besteht beim verwendeten Format. Zusätzlich lassen sich aber auch ASCII-Steuerzeichen eingeben. Das Sector-Modul bietet also sehr viel interessante Befehlssätze, doch damit ist noch lange nicht genug. Desweiteren kann eine Diskette sequentiell nach beschriebenen Blöcken abgesucht werden, Blöcke können in einen gewünschten Zielsektor geschrieben werden, der Hashwert von beliebigen Zeichenketten wird errechnet und auf das Feld mit der berechneten Indexnummer wird eine Schreibmarke gesetzt, eventuell fehlerhafte Prüfsummen können korrigiert werden und bestimmte Bytefolgen können auf der Diskette gesucht werden. Natürlich besteht auch die Möglichkeit, den Inhalt eines Sektors auf Drucker ausgeben zu lassen.

Medizin gegen Read/Write Error

Wie man also deutlich erkennt, ist dieses Sector-Modul sehr inhaltsreich und mit etwas Übung ist es kein Problem mehr, den gelöschten File-Header einer bekannten Datei oder den Dateianfang aufzufinden.

Track-Modul: Mit diesem Modul können die einzelnen Spuren der Diskette bearbeitet und korrigiert werden. Falls eine Diskette mit einem Read/Write-Error versehen ist, dann kann diese mit dem Track-Modul wieder instandgesetzt oder die Daten gerettet werden. Die wesentliche Voraussetzung, daß eine Diskette wieder verwendet werden kann ist, daß diese nicht physisch defekt ist. Der Inhalt einer zerstörten Spur wird eingelesen und kann dann auf eine andere Spur zurückgeschrieben werden. Falls das Symbol „Allow Change" angewählt wurde, kann der Inhalt der defekten Spur auch auf eine andere Zieldiskette zurückgeschrieben werden. Nachdem alle einwandfreien Sektoren auf der Sicherungsdisk abgespeichert sind, können diese mit dem Sector-Modul in einen vollständig funktionierenden Track umgebaut werden.

File-Modul: Hiermit ist es möglich, auf die Form, Funktion und den Inhalt der Dateien Einfluß zu nehmen. Desweiteren können Dateien, deren Header- und Listenblöcke noch intakt sind, wieder repariert werden, falls diese eine fehlerhafte AmigaDOS-Struktur aufweisen. Der gesamte Inhalt einer Datei wird in den Arbeitsspeicher geladen (Vorausssetzung ist genügend Speicher!) und kann anschließend mühelos bearbeitet werden. Wer mit dem Umgang des Sector-Moduls vertraut ist, der hat auch hier keine Schwierigkeiten, mit den wirkungsvollen Befehlen umzugehen und den Inhalt der Dateien zu ändern.

Bitmap-Modul: Jede Diskette enthält einen Blockbelegungsplan, eine Bitmap, die angibt, welche Blöcke belegt sind und welche frei sind. Einzelne Blöcke können gesperrt oder auch freigegeben werden. Desweiteren gewinnt man einen großen Einblick über die Verwendung der Blöcke unter AmigaDOS und besonders über die unbelegten Blöcke, um eventuell gelöschte Dateien wieder zu rekonstruieren.

Paßwortschutz erlaubt Sicherheitskopien

Alle Operationen werden im Pufferspeicher ausgeführt und erst auf Wunsch auf die Diskette zurückgeschrieben. Am besten ist es, „DISCovery" selbst auszuprobieren, um die Vielseitigkeit dieses Programmes einschätzen zu können. Der Autor verzichtet auf einen Kopierschutz, lediglich eine Paßwortabfrage ist eingebaut. Wer viel mit Disketten zu tun hat, sollte bei „DISCovery" zuschlagen, da es kein Programm für einen Abend ist, sondern ein tägliches Hilfsmittel darstellt. Somit geht auch der Preis völlig in Ordnung.

Info: DTM, Poststr. 25, 6200 Wiesbaden, Tel.: 06121/560084, Preis: 198 Mark

(Stephan Quinkertz)

Hochauflösend ohne Flickern

Farbenvielfalt, hohe Auflösung: All das kann Amiga. Allerdings mit einem schwindelerregenden Nachteil. Commodore nannte es Interlace, Freaks schlicht Flickern. Jetzt gibt es eine Lösung.

Entwicklungsdevise aus den Entwicklungsjahren des Amiga lautete: Das technisch Machbare mit ökonomischen Mitteln erreichen. Hohe Bildschirmauflösung war mit Hilfe zweier Halbbilder leichter zu erreichen, als mit einem Vollbild, da so die Datenübertragungsgeschwindigkeit beim Direct Memory Acces niedriger gewählt werden konnte, was weniger aufwendige Chips benötigte. Dafür aber nahm man das Flickern des Interlacemodus in Kauf, was zur damaligen Zeit akzeptabel erschien. Flickern schien da nur ein kleiner Nachteil. Jetzt ist es anders. Nahezu schrottreife PC-Oldtimer, früher belächelt, ziehen mit VGA-Grafik und Multisync-Monitor am Amiga vorbei — ohne Interlace.
Auf der CeBIT zeigte Commodore jetzt in Verbindung mit der neuen Amiga-2500-Serie neue Customchips, die in der höchsten Auflösung nicht mehr flickern. Dafür büßt der Anwender in diesem neuen Grafikmodus die Farbenpracht des alten Interlacemodus ein: Nur noch vier Farben werden dem Benutzer angeboten.
Innovativen jungen Firmen dienen solche kleine Schwächen als ökonomische Nischen. Auf der letzten CeBIT konnte man das Ergebnis echter Bemühungen um die Probleme der User sehen: Der FlickerFixer von MicroWay macht den Interlacemodus flimmerfrei — in allen möglichen Farben (also auch Interlace Hold and Modify). Die trickreiche Hardware für den Amiga überlistet das Videosignal mit einer einfachen Methode. In 256K superschnellem Static-RAM wird das Videobild zwischengespeichert und mit Hilfe von Videosteuerungsbausteinen auf doppelte Bildwiederholfrequenz gebracht. Das heißt, wenn die Amiga-Videochips das eine Halbbild an den Monitor absenden, speichert der FlickerFixer das Bild intern ab, und während der Amiga das nächste Halbbild sendet fügt der Framebuffer das neue Bild mit dem alten zusammen. Daraus ergibt sich eine effektive Bildwiederholfrequenz für ein Vollbild von vollen 50 Hertz für die PAL Version der Amigas.

Mit neuer Karte fit in jedem Grafikmodus

Der FlickerFixer arbeitet aber nicht nur im Interlacemodus, sondern er unterstützt jeden Grafikmodus des Amiga mit dem vollen Overscan von bis zu 702 mal 564 Punkten PAL. Umstöpseln des Monitors oder zwei verschiedene Monitore sind nicht nötig.
Die Platine wird im Videoslot des Amiga 2000 eingesetzt, benötigt aber den langen Videoslot des Typ B Motherboards. Amigas mit dem B-Typ Motherboard sind äußerlich am gelben Chinch-Video-Anschluß neben den beiden Chinch-Audiobuchsen zu erkennen. Der Videokontroller des FlickerFixer sendet das Videosignal aber nicht mehr mit der Zeilenfrequenz von 15,75 kHz sondern mit 31,5 kHz. Das heißt, daß der FlickerFixer nicht mehr mit einem normalen Amigamonitor zusammenarbeitet, sondern einen IBM-VGA-Grafik-kompatiblen oder einen multisynchronen Monitor zum Anschluß mit einem Standardkabel benötigt. Schon im normalen Grafikmode fallen die Qualitäten dieser Monitore auf: das Bild ist wesentlich besser als beim üblichen Amigamonitor, schärfer und ruhiger. Die Farben sind viel satter und unglaublich brillant. Ferner sollte getestet werden, ob die Verbindung Amiga und Multisync auch ohne FlickerFixer schon eine Verbesserung der Bildschirmdarstellung mit sich bringt.
Mit einem speziellen Kabel (Anschlußbelegung in Bild 2) startete die zweite Runde des Tests. Hier bewies der Multisync, daß er für den größten Teil der Bildverbesserungen verantwortlich ist. Interlace allerdings flickerte wie es der User schon immer gewöhnt ist. Die Leistung des FlickerFixers liegt ausschließlich in der Beseitigung des Interlacemodus.
Alle Programme (Rundumschlag quer durch alle Spiele und Anwenderprogramme) liefen mit der Grafikkarte. Lediglich Hardwareerweiterungen wie ein Genlock-interface laufen mit dem FlickerFixer nicht ordnungsgemäß zusammen. Einen genauer Test aller auf dem Markt befindlichen Grafik-

KABELBELEGUNG für einen Multisync an den Amiga Videoanschluß:

AMIGA		MONITOR
BUCHSE 23 Pol Sub D Stecker Stecker		9 Pol Sub D
PIN	3 ... Analog Rot	PIN 1
	4 ... Analog Grün	2
	4 ... Analog Blau	3
	10 ... Composite Sync	4
	16-19 ... Erde	6—9

hardware-Erweiterungen zusammen mit dem FlickerFixer konnte im Rahmen dieses Tests nicht durchgeführt werden. Erwähnenswert: das normale Amiga- RGB-Signal steht weiter zur Verfügung. Ein multisynchroner Monitor für sich allein bietet schon ein wesentlich verbessertes Bild am Amiga und eine deutliche Schonung von Augen und Nerven. Zumal es im Wesen eines Multisyncs liegt, auch mit vielen anderen Grafikstandards der PC-Welt zusammenzuarbeiten (Megavision 01 oder 02 von Sang, einem Grafiksubsystem für den Amiga oder mit jeder IBM-PC kompatiblen Grafikkarte). Er ist also eine Anschaffung für jede Gelegenheit.

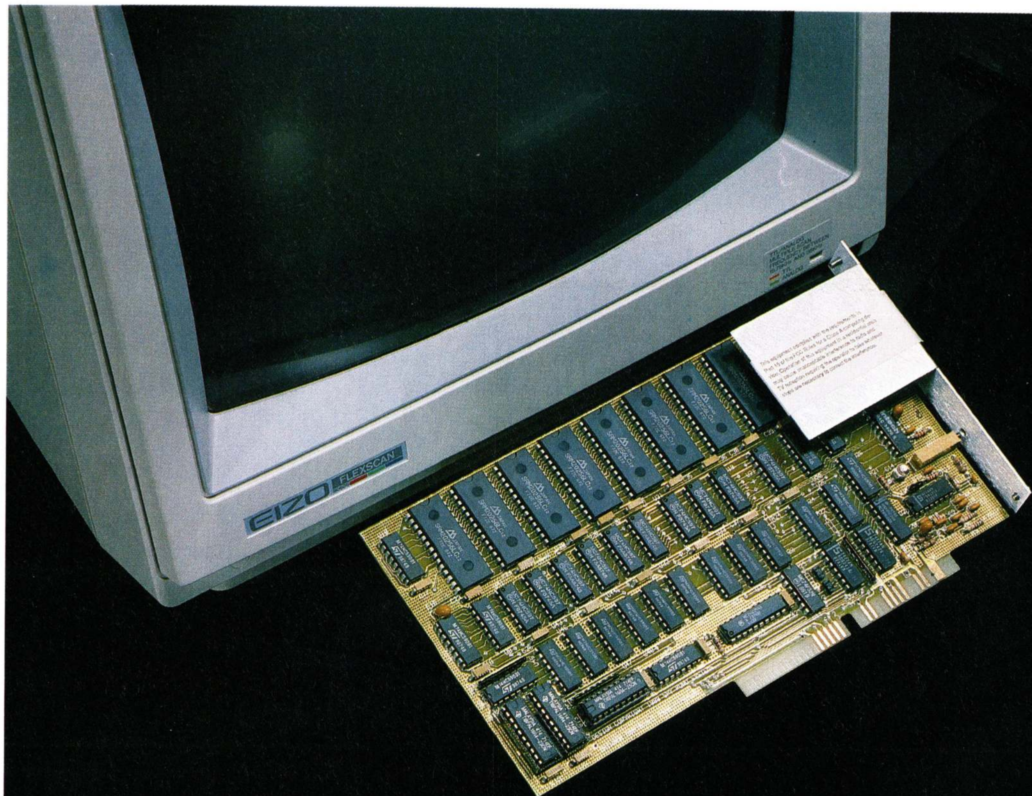
Standfestes Bild hat seinen Preis

Ein Multisync mit einem FlickerFixer stellt momentan das Non-plus-ultra für die Bildschirmgrafikformate des Amigas dar.

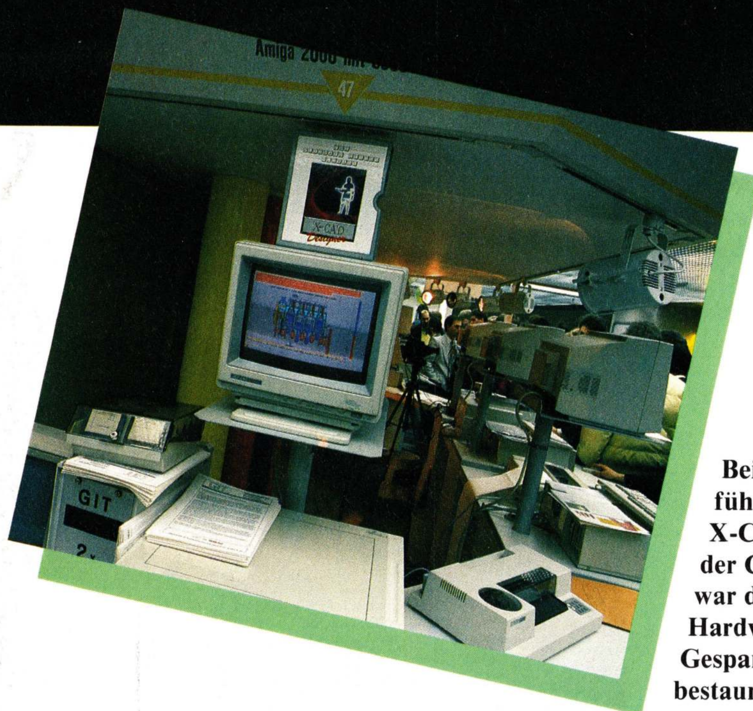
Nach soviel Lob ein Wermutstropfen: FlickerFixer in Verbindung mit einem Multisync ist teuer. Für Hobbyanwendungen nahezu unbezahlbar, denn die Kombination wiegt nahezu einen gut ausgestatteten Amiga 2000 ohne Monitor auf.

(Wilfried Häring)

Info: GIT-Wiese, Maassenstr. 10, 4235 Scharnbeck, Tel.: 0 28 53/40 99, Preis: 10 395 Mark, Komplettangebot FlickerFixer plus EIZO Flexscan 8060 S: 2995 Mark



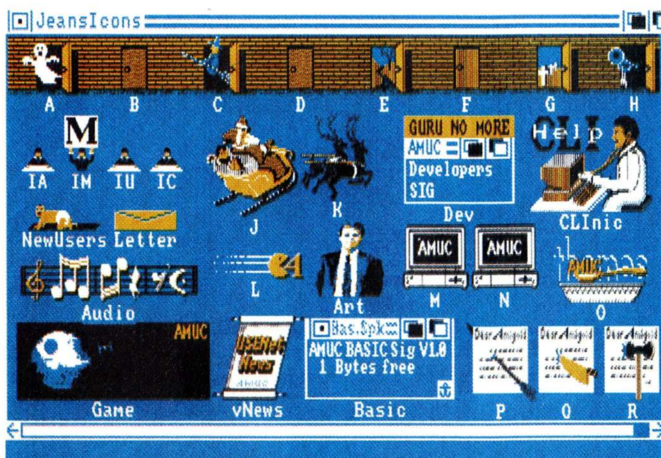
FlickerFixer und Multisync im Duo: So verliert Interlace seinen Schrekken.



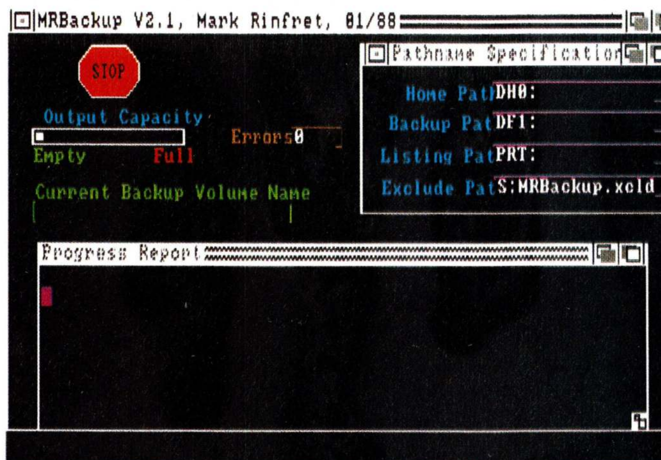
Bei der Vorführung von X-Cad auf der CeBIT war das Hardware-Gespann zu bestaunen.

Fangfrisch! FISHWARE 128 – 138

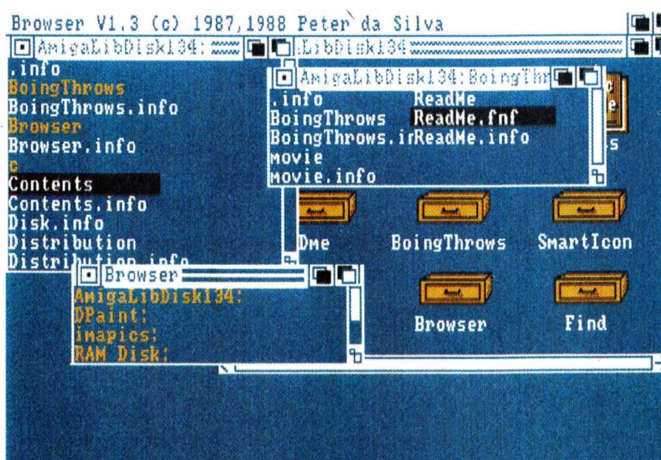
Während man sich noch durch die letzte Fishdisk arbeitet, sorgt Mr. Public Domain schon wieder für Nachschub. PD-Software vom Fließband?



Großer Icon-Markt: Mit dem Icon-Assembler von AmigaWelt lassen sich die Beispiele aus Fish 137 überall verwenden



Unverzichtbare Datensicherheit der Festplatte durch Backup: auf Fish 129 erhältlich



Wer Icons nicht mag: Browser bietet gleichen Service auch ohne Schnickschnack

Ständig kommen neue PD-Programme auf den Markt — vom Absinken des Niveaus kann jedoch keine Rede sein. Hervorragende Programme befinden sich im neuesten Disketten-Container. Die persönliche Handschrift, die Fred Fish seinen Disketten schon durch die interessante und abwechslungsreiche Programmzusammenstellung verleiht, wird durch kleine Anekdoten unterstrichen. So, wenn er von einem Absturz seines Amiga berichtet — verbunden mit der Verabschiedung eines Quellcodes, der auch uns aus diesem Grunde vorenthalten bleiben muß.

Angesichts der vielen neuen Programme ist dieser Verlust jedoch leicht zu verschmerzen. Neben Updates zu bekannter und bewährter Software finden sich wieder viele neue Anwendungsprogramme, Utilities, Programmierhilfen und Demos.

Auf den brandneuen Fishdisks 128—138 überwiegt allerdings wieder die Benutzerführung über das CLI. Ein Grund mehr, sich einmal näher mit dieser Benutzerschnittstelle auseinanderzusetzen, die zwar schwieriger zu bedienen, aber auch wesentlich leistungsfähiger ist als die Workbench. Zugegeben, der fehlende Editierkomfort des CLI ist schon ein Ärgernis, aber auch hier kann die Fishware weiterhelfen: Conman, eine hervorragende Editierhilfe von Fishdisk 133, die am Ende dieses Artikels genauer besprochen werden wird.

Fishdisk 128 platzt mit neun Programmen aus allen Nähten. „Dis" ist ein leistungsstarker DisAssembler mit Quellcode. Zwei Harddisk-Backup-Programme, eine neue Uhrenvariante (LED-Clock) erhält der User, ebenso wie die Nachbildung eines Unix-Stream-Editors. „Wkeys" ermöglicht Manipulationen von Fenstern mittels Tastatur, „DropCloth" das Unterlegen des Workbenchscreens mit IFF-Bildern.
Fishdisk 129 enthält mit „MRBackUp" (Version 2.0 und 2.1) Updates zu der Version auf Disk 128. Es handelt sich um ein sehr flexibles, workbenchgesteuertes Festplattenbackup-Programm, welches optional auch eine Datensicherung in komprimierter Form erlaubt. Eines der ganz speziellen Features läßt sich glücklicherweise abschalten. MRBackUp nutzt nämlich die Sprachsynthese des Amiga und plappert bei sämtlichen Arbeitsschritten munter drauflos. „DosKwick" 2.0 besteht aus zwei Programmen, RiteKwick und ReadKwick, die ein etwa dreimal schnelleres Laden und Speichern von Daten erlauben. Der Zeitgewinn wird durch den Nachteil erkauft, daß die Daten nicht im AmigaDOS-Format gespeichert werden. „PaintJet" ist ein direkt aus dem Hause Hewlett-Packard stammender Druckertreiber für den HP-Tintenstrahldrucker gleichen Namens. „Patch" 1.3 und 2.0 sind Amiga-Implementationen des gleichnamigen Unix-Befehls, die zusammen mit dem Befehl „Diff" von Fish 138 die Möglichkeit zu einem schnellen Updaten von Dateien liefern.
Zur Anschaffung von Fishdisk 130 kann allein schon wegen „QMan" geraten werden. Dieser Mandelbrotgenerator läßt sich einfach über Menüs bedienen und erzeugt wunderschöne „Apfelmännchen". „Evo" veranschaulicht die unterschiedlichen Stadien der menschlichen Evolution und „Hp" simuliert einen Taschenrechner einschließlich trigonometrischer Funktionen. „Mach" ist ein Mausbeschleuniger mit zusätzlichen Features wie HotKeys, ClicktoFront oder PopCLI. Mit „DirMaster" 1.1 liegt ein Update des Diskettenkatalogisierers von Disk 108 vor. Grafische Muster lassen sich auf einfache Weise mit „PatEdit" erstellen.
Besonders vielseitig ist die Mischung auf Fishdisk 131. „Dfc", ein multitaskingfähiges Kopierprogramm, erlaubt bei genügendem Speicherplatz beliebig viele Kopierdurchgänge ohne erneutes Auslesen der Quelldisketten. „Hyperbase" 1.6 ist ein Update zum Datenbankprogramm auf Diskette 58, Life eine neue Version der bekannten Zellwachstumssimulation. „Wfrags" informiert in einem Fenster, das ständig aktualisiert wird, über die momentane Speicherbelegung, während sich mit „Mg1b" sämtliche Tasten mit Makros belegen lassen. „Mackie" ist eine nette Variante zu PopCLI. Die integrierte Funktion zum Schonen des Monitors begnügt sich nicht damit, nach längerer Inaktivität des Benutzers den Bildschirm dunkel zu schalten, sie füllt ihn stattdessen mit schönen geometrischen Figuren.
Zunächst einen etwas kargen Eindruck macht Fishdisk 132. Sie enthält nur die Demo „Berserk", für die der Autor Leo L. Schwab beim FAUG-Treffen im September '87 stehende Ovationen bekam. Was man nicht auf den ersten Blick sieht: Die Diskette enthält neben dem sehenswerten Resultat sämtliche Quelldateien zu diesem Demo. Fred fand es einfach an der Zeit, „to have at least one animation that was available at the source code level."
Neben der noch zu besprechenden Version 1.1 von „Conman" offeriert Fishdisk 133 ein Extrabonbon für europäische User. Wer sich über Programme ärgert, deren Fenster sich nicht über das NTSC-Format hinaus vergrößern lassen, kann mit „Overscan" die Intuition.library patchen und dieses Problem aus der Welt schaffen. „Crc" dient dem schnellen Verifizieren von Dateiinhalten über 16-bit-crc's (cyclic redundancy checks). Zu diesem Programm liefert Fred mit „Crclists" alle „Checklisten" für seine Disketten bis Nr. 128. Damit kann jeder schnell überprüfen, ob die eigene Fish-Sammlung mit Freds Masterdisketten übereinstimmt.
„BoingThrows" auf Disk 134 ist eine sehr schöne HAM-Animation, bestehend aus 50 Einzelbildern, deren Berechnung 325 Stunden dauerte. Kein Wunder, daß der Autor sich ganz offen zu seinem sehnlichen Wunsch nach einer Turbo-Karte bekennt. Mit „Browser" erhält man eine Benutzeroberfläche, die der Workbench stark ähnelt, nur werden statt der Icons die Dateinamen angezeigt, die auf gleiche Weise wie die gewohnten Piktogramme anzuklicken und zu behandeln sind.

Mit „find" auf der Suche

Mit Browser lassen sich bequem auch die Dateien bearbeiten, die nicht über eigene Icons verfügen. „Dme" 1.29 ist die aktuelle Version eines Editors, der speziell auf die Belange von Programmierern zugeschnitten ist. „Library" erlaubt platzsparendes Speichern und rasches Wiederauffinden von Textdateien, während „find" dem Herausfiltern von Dateien nach Kriterien wie Name, Dateigröße oder Zeitpunkt der letzten Änderung dient.
TeX-Anwender finden auf Fishdisk 135 insgesamt 78 Zeichensätze verschiedener Größe bis hin zu über 150 Pixeln. Als Zugabe gibt es „TeXF", ein Programm zur Konvertierung von PK-Fonts (das heißt: komprimierten TeX-Fonts) in Amiga-Zeichensätze. Konvertieren lassen sich jedoch nur die ASCII-Zeichen von 32 bis 127, im übrigen sind auch die Einsatzmöglichkeiten der konvertierten Zei- ▶

chensätze beschränkt. Auch mit TeX's Zeichensätzen ausgestattet, kann Amiga-DOS diesem professionellen Schriftsatzprogramm in keiner Hinsicht das Wasser reichen.

Der Inhalt von Fishdisk 136 wird in erster Linie von Updates bestimmt. „IFF2Pcs" ist die neueste Version des Puzzle-Programms von Disk 122. „YaBoingII", ein kleines Spiel zur Demonstration von Hardware-Sprites und Kollisionsabfrage, geht auf die Fassung von Disk 36 zurück, während mit „Zoo" die Version 1.71 eines Programms zum Komprimieren und Archivieren von Dateien zur Verfügung gestellt wird. Abhängig von der inneren Struktur der zu komprimierenden Dateien erreicht das verwendete Lempel-Ziv-Verfahren eine Platzersparnis von 20—80 Prozent. Mit „AsmToolBox" erhalten Assembler-Programmierer nützliche Utilities, Unix-Freunde hingegen mit „Paste" und „Bison" die Amiga-Implementationen der Befehle Paste und yacc.

Piktogramm-Fetischisten können auf Fishdisk 137 eine Reihe einfallsreicher „JeansIcons" bestaunen und selbstverständlich auch nutzen. C-Programmierer können mit „Vgad" eigene Gadgets aus IFF-Bildern erstellen. Das Programm erzeugt den entsprechenden Quellcode. Im Gegensatz hierzu erfordert „Sit", ein Utility zum Ändern des Icontyps, keine Programmierkenntnisse. Der Knüller auf dieser Diskette dürfte aber „Ct" sein, ein Programm, welches die Bilder eines Computertomographen auf dem Amiga darstellt und eine Änderung ihrer Kontrast- und Farbwerte gestattet. Da dieses Programm einschließlich der mitgelieferten Beispielbilder eine gesamte Diskette beanspruchen würde, wurde es mit Zoo komprimiert. Mit diesem Programm muß die Datei ct.zoo erst dekomprimiert werden, um eine lauffähige Version zu erhalten. Zwei weitere Programme „bespitzeln" die Laufwerke von Amiga. Während „Muncho" jedes Einlegen oder Entnehmen einer Diskette mit gurgelnden Lauten quittiert, erfüllt „VirusX" eine sehr sinnvolle Aufgabe: Der Bootblock jeder neu eingelegten Diskette wird automatisch überprüft und sobald er von der üblichen Form abweicht, wird eine Warnung ausgegeben.

Neben dem schon erwähnten Programm „Diff" präsentiert Fred Fishs vorerst letzte Diskette ein Programm zur Konvertierung von Mac-Fonts in Amiga-Zeichensätze und zwei neue Versionen der praxisbewährten Terminalemulation „Vt100". Für Programmierer gedacht sind „ModulaTools", ein Update der Version auf Disk 94, und „AmigaLine", eine Fülle von Programmtips und Informationen aus dem Usenet.

Conman — Editierhilfe für das CLI

Man steht vor einer schwierigen Entscheidung, wenn man aus so vielen hervorragenden Programmen ein einzelnes hervorheben soll. Obwohl es nicht mehr ganz neu ist, habe ich mich diesmal für Conman von Fishdisk 133 entscheiden, da es die Arbeit mit dem CLI erheblich erleichtert und somit für jeden Amgia-Anwender interessant sein dürfte. Conman ist eine äußerst komfortable Editierhilfe für das CLI, wirkt aber auch auf alle anderen Fenster, die über das con:-Gerät geöffnet werden. Bevor man dieses Programm nutzen kann, muß es installiert werden. Zwar verheißen zwei Icons in der „Conman"-Schublade ein einfaches Starten, und auch der Begleittext fordert zu einem munteren Anklikken dieser Symbole auf, mehr als eine Fehlermeldung kommt auf diese Weise jedoch nicht zustande. Der Grund: Conman sucht das Programm „Conhandler" in der „L"-Schublade; die aber ist auf der Fishdisk gar nicht vorhanden. Am besten ist es, alle drei erforderlichen Dateien aus der „Conman"-Schublade auf die Workbench-Diskette zu kopieren, und zwar die Datei „Conman" in das „c"-Directory, die Datei „Conhandler" in das „L"-Directory, sowie „conhandler.library" in das „Libs"-Directory. Falls Sie über zwei Laufwerke verfügen, sich die Workbench in df0: befindet und die Fishdisk 133 in dfx:, dann sieht das konkret so aus:

```
1> cd dfx:conman
1> copy conman to df0:c
1> copy conhandler to
     df0:L
1> copy conhandler.
     library to df0:Libs
```

Sie haben nur ein Laufwerk? Dann sollten Sie zum Arbeiten mit AmigaDOS ohnehin alle erforderlichen Befehle in die Ramdisk kopieren und das c-Directory neu zuweisen. Die folgenden Befehle setzen voraus, daß Ihre Workbench in df0: liegt.

```
1> cd df0: 1 makedir ram:c
1> copy c to ram:c all
1> assign c: ram:c
```

Ersetzen Sie die Workbench durch die Fishdisk:

```
1> makedir ram:conman
1> copy df0:conman ram
     :conman all
```

Nun legen Sie wieder die Workbench ein:

```
1> cd ram:conman
1> copy conman to df0:c
1> copy conhandler to
     df0:L
1> conhandler.library to
     df0:Libs
```

Nach dieser Installation rebooten Sie bitte Ihr System mit der soeben erweiterten Workbench und gehen in das CLI. Aktivieren Sie Conman, indem Sie

```
1> conman -c
```

eingeben. Conman wirkt auf alle Fenster, die von nun an neu geöffnet werden. Geben Sie „Newcli" ein und erproben Sie in diesem Fenster Conmans Fähigkeiten. Tippen Sie einige Zeichen ein, und probieren Sie anschließend die Tasten „Cursor left", „Cursor right", „Backspace" und „Del". Sie funktionieren, wie man es von Textverarbeitungen gewohnt ist und erlauben ein schnelles Verbessern von Tippfehlern. Geben Sie nun hintereinander einige Befehle ein, und versuchen Sie die „Cursor up"- und „Cursor down"-Tasten. Wie Sie sehen, speichert Conman Ihre Eingaben in chronologischer Folge und läßt sie auch wieder abrufen. Wenn AmigaDOS zum Beispiel einen Befehl wegen eines Eingabefehlers verweigert, genügt ein Tastendruck und die gesamte Zeile steht zum Verbessern bereit. Mit den Funktionstasten lassen sich unter anderem ganze Worte löschen, das aktuelle Fenster auf Minimalgröße oder auf volles Bildschirmformat bringen, in den Vorder- oder Hintergrund schalten und vieles mehr. Sämtliche Fähigkeiten Conmans aufzuführen, würde den Rahmen dieses Artikels „geringfügig" sprengen.

(Ralf Leithaus)

Jet mit deutscher Anleitung ist da! nur DM 99,90

Deutsche Anleitung einzeln nur DM 15,90

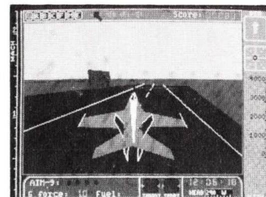
Zögern Sie nicht, kaufen Sie sich Ihren persönlichen Jet-Simulator.

Luft-Luft/Luft-Boden/Luftbodenkampf wählbar.

Nachnahme 6,— DM, Vorkasse 4,— DM, Ausland: nur gegen Vorkasse 10,— DM

PDC GmbH, Louisenstraße 115, 6380 Bad Homburg
Telefon (0 61 72) 2 47 48 oder 2 07 99





Prg. für alle AMIGA-Modelle
– Exzellent in Struktur, Grafik, Sound – alle Prg. in Deutsch –

ASTROL. KOSMOGRAMM
– Nach Eingabe von Namen, Geburtsort (geografischer Lage) und Geburtszeit werden errechnet: Sternzeit, Aszendent, Medium Coeli, Gestirnestände im Tierkreis, Häuser nach Dr. Koch/Schäck (Horoskop-Daten mit Ephemeriden. Außer dem Bildschirmdisplay kann Ausdruck auf 2 DIN A4-Seiten erfolgen; davon 1/2 Seite allgemeines Persönlichkeitsbild mit Partnerschaftskriterien und 1/2 Seite Tierkreisdiagramm (Horoskop). Alle Planeten mit Sonne und Mond. Für alle Berufs- und Hobby-Astrologen eine unentbehrliche Arbeitserleichterung. **78,–**

BACKGAMMON **68,–**

BIOKURVEN
Zur Trendbestimmung der Biorhythmen und des seelisch-/geistig-/körperlichen Gleichgewichts mit Druck des Kurvendiagramms von oben nach unten in beliebiger Länge.
In der rechten Blatthälfte das Diagramm, links eine Auswertung des Gesamtpotentials für jeden Tag. Werte für bestimmte Tage auch auf dem Bildschirm. Ausführliche Beschreibung der wissenschaftlichen Grundlagen.
Ideal für Partnervergleiche. **58,–**

GESCHÄFT
– Bestellung, Auftragsbestätigung, Rechnung, Lieferschein, Mahnung, 6 Briefrahmen mit Firmendaten zur ständigen Verfügung (Anschrift, Konten usw., Menge/Preis, Rabatt/Aufschlag, MwSt., Skonto, Verpackung, Versandweg usw.) Mit Einbindung von abgespeicherten Adressen und Artikeln. **198,–**

GELD
– Man wählt mit der Maus unter 25 Rechenroutinen in den Bereichen: Anlage – Kapital – Vermögensbildung – Rentensparen – Rendite – Lasten – Zinsen/Zinseszinsen – Kredit – Hypotheken – Laufzeit – Amortisation – Ratenzahlung – Wertverlust – Nominal- und Effektivzinsen – Ausdruck vollständiger Tilgungsraten – Diskontierung – Devisen/Sorten – Konvertierung **98,–**

KALORIEN-POLIZEI – Nach Eingabe von Größe, Gewicht, Geschlecht, Arbeitsleistung erfolgt Bedarfsrechnung und Vergleich m. d. tatsächlichen Ernährung (Fett, Eiweiß, Kohlehydrate), Idealgewicht, Vitalstoffe, auf Wunsch Ausdruck. **58,–**

usw. usw. – Fordern Sie mit Freiumschlag unsere Liste an!
Im Computer-Center oder bei uns zu obigen unverbindlich empfohlenen Preisen + DM 3,– bei Vorkasse oder DM 4,70 bei Nachnahme



I. Dinkler
Am Schneiderhaus 17 · D-5760 Arnsberg 1
Tel. 0 29 32 / 3 29 47

Dateiverwaltungen:

Personal	98,—
Adressen	69,—
Bibliothek	88,—
Diskothek	78,—
Dokumente	98,—
Exponate	118,—
Galerie	118,—
Inventur, Fibu-gerecht	118,—
Lagerartikel	88,—
Museum	118,—
Videothek	78,—

Wenn mal was nicht funktioniert …

Computer-Service

commodore
Schneider
ATARI

Installation
Wartung
Reparatur



Technischer Kundendienst

Im gesamten Bundesgebiet vertreten
70 Niederlassungen
Hotline Bereich Nord (040) 2201913
Hotline Bereich Mitte (0201) 35923
Hotline Bereich Süd (08165) 74220
Hotline Berlin (030) 684 60 57–9

KREUZ AS
Amiga Roulette
SOFTWARE 2000
AS
Bundesliga Manager

Lange Straße 51,
2320 Plön / Holstein
Telefon 0 45 22 / 13 79

Weitere AMIGA - SOFTWARE in Vorbereitung! Händleranfragen erwünscht! Alle Programme laufen auf allen AMIGA - Modellen! Bestellungen schriftlich oder telefonisch unter **0 45 22 / 13 79** · Gegen 1,30 DM in Briefmarken erhalten Sie ausführliche Produkt-Info's Versand gegen Vorkasse oder per Nachnahme zuzüglich 5,-- DM für Porto und Verpackung.

KREUZ AS
– Sehr guter Poker-Spielautomat
– Nach dem Original-Spielhallen-
– Hit „Herz As"
– Sehr gute Grafik und
– dig. Sound.
Best.-Nr. S 02 002 88
Preis: 79.– DM

AMIGA ROULETTE
– Tolle Grafik
– bis zu 4 Spieler
– hoher Spielspaß
– alle Roulette-Regeln werden
– berucksichtigt
– dt. Anleitung
Best.-Nr.: S 01 001 88
Preis: 69.– DM

BUNDESLIGA - MANAGER
– Werden Sie zum Manager
– Ihres Fußballvereins.
– Verhelfen Sie Ihrem Verein zu
– Siegen und Geld.
– Sehr gute Wirtschafts-
– Simulation
– hoher Spielspaß
– bis zu 4 Spieler
Best.-Nr.: S 02 001 88
Preis: 69.– DM

NEUE AMIGA - SOFTWARE

Ecosoft Economy Software AG

Postfach 1905, 7890 Waldshut, Tel. 07751 - 7920

Software (fast) gratis

- **Größtes Angebot von Prüf-Software und Frei-Programmen Europas**: Über 3'000 Disketten für IBM-PC/ Kompatible, Macintosh, Amiga, Atari ST, C64/128, Apple II.
- Viele **deutsche Programme**, speziell für Firmen, Selbständige, Privatpersonen, Schulen.
- **Software gratis**, Sie bezahlen nur eine Vermittlungsgebühr von DM 14.40 oder weniger je Diskette.
- HOTLINE: Tel. Anwenderunterstützung (3 Techniker von 8-18 Uhr).

Verzeichnis gratis

Bitte Computermodell angeben. Gegen Einsendung dieses Inserates erhalten Sie zusätzlich einen Gutschein für eine Gratis-Diskette.
427

Bibliotheken: Parameter und Funktionen Teil 1

EXEC*		A0	A1	A2	A3	D0	D1
AbortIO	-480		IORequest				
AddDevice	-432		Device				
Addhead	-240	List	Node				
AddIntServer	-168		Interrupt			Intnumber	
AddLibrary	-396		Library				
AddPort	-354		Port				
AddResource	-486		Resource				
AddTail	-246		List	Node			
AddTask	-282		Task	initPC	finalPC		
AllocAbs	-204		localTOn			Size	
Allocate	-186	freelist				Size	
AllocEntry	-222	Entry	pvsw0080sc00pxpv				
AllocMem	-198					Size	Request
AllocSignal	-330					Signal-Num	
AllocTrap	-342					TrapNum	
AvailMem	-216						request
Cause	-180		Interrupt				
CheckIO	-468		IORequest				
CloseDevice	-450		IORequest				
CloseLibrary	-414		Library				
Deallocate	-192	FreeList	MemBlock			Size	
Debug	-114						
Disable	-120						
Dispatch	- 60						
DoIO	-456		IORequest				
Enable	-126						
Enqueue	-270	List	Node				
Exception	- 66						
ExitIntr	- 36						
FindName	-276	List	Name				
FindPort	-390		Name				
FindResident	- 96		Name				
FindTask	-294		Name				
Forbid	-132						
FreeEntry	-228	Entry					
FreeMem	-210		MemBlock			Size	
FreeSignal	-336					SignalNum	
FreeTrap	-348					TrapNum	
GetCC	-528						
GetMsg	-372	Port					
InitCode	- 72					StartClass	Version
InitResident	-102		Resident				SegList
InitStruct	- 78		InitTable	Memory		Size	
Insert	-234	List	Node	Pred			
MakeFunctions	- 90	Target	FuncArray	FuncDispBase			
MakeLibrary	- 84	funcInit	Structinit	LibInit		DataSize	CodeSize
OldOpenLibrary	-408		LibName				
OpenDevice	-444	DevName	IORequest			Unit	Flags
OpenLibrary	-552		LibName			Version	
OpenResource	-498	ResName				Version	
Permit	-138						
Procedure	-540	SemaPort	BidMsg				
PutMsg	-366	Port	Message				
RawIOInit**	-504						
RawMayGetChar**	-510						

RawPutChar**	-516			Char	
RawDoFmt**	-522				
RemDevice	-438		Device		
RemHead	-258	List			
RemIntServer	-174		Interrupt	IntNumber	
RemLibrary	-402		Library		
Remove	-252		Node		
RemPort	-360		Port		
RemResource	-492		Resource		
RemTail	-264	List			
RemTask	-288		Task		
ReplyMsg	-378		Message		
Reschedule	- 48				
Schedule	- 42				
SendIO	-462		IORequest		
SetExcept	-312			NewSig	SignalSet
SetFunction	-420	FuncOffset	Library	FuncEntry	
SetIntVector	-162		Interrupt	IntNum	
SetSignal	-306			NewSig	SignalSet
SetSR	-144			NewSR	Mask
SetTaskPri	-300		Task	Priority	
Signal	-324		Task	SignalSet	
SumLibrary	-426		Library		
SuperState	-150				
Supervisor	- 30				
Switch	- 54				
TypeOfMem	-534		Adress		
UserState	-156			SysStack	
Vacate	-546	SemaPort			
Wait	-318			SignalSet	
WaitIO	-474		IORequest		
WaitPort	-384	Port			

DOS		D1	D2	D3	D4
Close	- 36	File			
CreateDir	-120	Name			
CreateProc	-138	Name	Priority	SegList	Stack-Size
CurrentDir	-126	Lock			
DateStamp	-192	Date			
Delay	-198	Timeout			
DeleteFile	- 72	Name			
DeviceProc	-174	Name			
DupLock	- 96	Lock			
Examine	-102	Lock	FileInfoBlock		
Execute	-222	String	File	File	
Exit	-144	ReturnCode			
ExNext	-108	Lock	FileInfoBlock		
Info	-114	Lock	ParameterBlock		
Input	- 54				
IoErr	-132				
IsInteractive	-216	File			
LoadSeg	-150	Filename			
Lock	- 84	Name	Type		
Open	- 30	Name	AccessMode		
Output	- 60				
ParentDir	-210	Lock			
Read	- 42	File	Buffer	Length	
Rename	- 78	OldName	NewName		
Seek	-66	File	Position	Offset	
SetComment	-180	Name	Comment		
SetProtection	-186	Name	Mask		
UnLoadSeg	-156	Segment			
Unlock	- 90	Lock			
WaitForChar	-204	File	Timeout		
Write	- 48	File	Buffer	Length	

* in dieser Ausgabe nur die 1.1 Exec-Routinen

** Routinen, die nur intern verwendet werden können

Was man über Drucke

Das Problem: Die richtige Druckerwahl. Die Hilfe: Grundlagen und Tips, um im Dschungel des Druckermarktes das Passende herauszufinden. Wir informieren über die Praxis der Druckeransteuerung, über die Leistungsmerkmale der unterschiedlichen Druckertypen und andere Gesichtspunkte, die beim Druckerkauf zu beachten sind.

Unbedingte Voraussetzung für den Betrieb eines Druckers am Amiga ist die richtige hardwareseitige Verbindung. Wie jeder Computer stellt auch Amiga für den Datenaustausch mit Peripheriegeräten sogenannte Schnittstellen zur Verfügung, die bestimmte Normen hinsichtlich des Datenübertragungsverfahrens erfüllen müssen. Ein problemloser Anschluß eines Peripheriegerätes ist nur dann möglich, wenn es über denselben Schnittstellentyp verfügt, wie der Rechner selbst.

In anderen Fällen ist zur Anpassung das Zwischenschalten eines Interfaces notwendig, was bei Amiga glücklicherweise so gut wie nie eintreten wird, da er mit den beiden auch auf Druckerseite häufigsten Schnittstellen ausgestattet ist: einer parallelen Centronics- und einer seriellen RS232-Schnittstelle. Insbesondere die Centronics-Schnittstelle, die sich bei jedem PC findet, ist weit verbreitet. Es gibt kaum einen Druckertyp, der nicht mit ihr erhältlich wäre. Dies ist auch schon der wichtigste Punkt beim Druckerkauf:

Er muß über eine dieser beiden Schnittstellen verfügen. Soweit möglich, sollte hier dem Centronics-Port der Vorzug gegeben werden, damit der serielle Port für andere Geräte wie Midi-Interfaces und Akustikkoppler freibleiben kann.

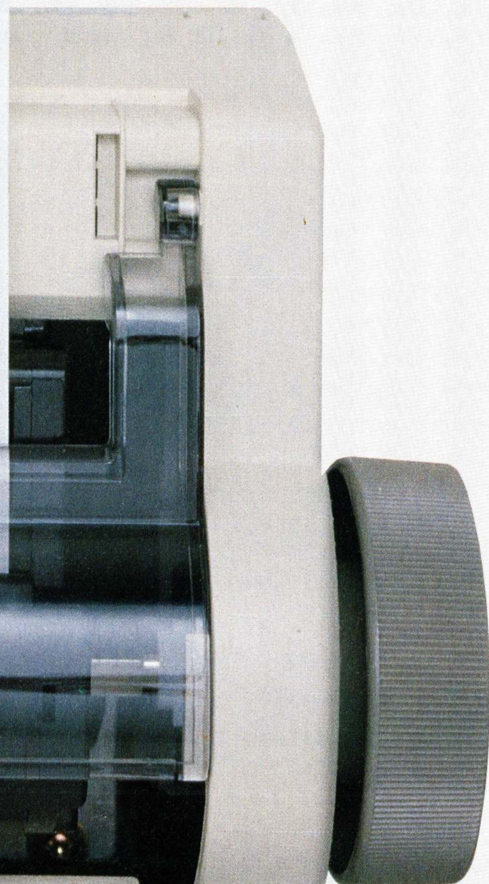
Aufsteiger vom C64 möchten vielleicht ihren alten Drucker weiterverwenden. Da die meisten dieser Geräte ausschließlich mit einem seriellen IEC-Bus ausgestattet sind, wird die Anschaffung eines recht teueren Interfaces nötig. Ob sich eine solche Ausgabe bei den stark gefallenen Druckerpreisen lohnt, wird nicht zuletzt auch von der verbliebenen

r wissen sollte

Lebenserwartung des alten Druckers abhängig sein. Einige „C64-Drucker“ erlauben eine interne Umrüstung auf Centronics durch ein Auswechseln des Schnittstellenmoduls. Ob dies möglich ist, erfahren Sie aus dem Druckerhandbuch oder von Ihrem Fachhändler.

ESC „?“ Druckerkauderwelsch schafft klare Verhältnisse

Der grundsätzliche Unterschied zwischen beiden Amiga-Schnittstellen liegt in der Übertragungsart. Wie für den Computer selbst, besteht auch für den Drucker die kleinste Informationseinheit aus einem Bit, umfangreichere Informationen aus einer mehr oder weniger langen Folge von Bits, die jeweils den Wert 0 oder 1 annehmen können. Um diese Informationen auswerten zu können, teilt der Drucker die Bitfolge in Gruppen von jeweils acht Bits, einem Byte, ein. Ein Byte kann 256 verschiedene Werte (von 0 bis 255) annehmen, und dies ist genau die Anzahl der Befehle die der Drucker versteht. Die meisten dieser Werte veranlassen den Drucker, ein Zeichen auszugeben, so sorgt zum Beispiel der Wert 65 dafür, daß auf dem Papier ein „A“ erscheint. Andere Werte stellen Steuerbefehle dar, beispielsweise bewirkt ein Wert von 12 einen Vorschub auf die nächste Seite. Eine besondere Bewandtnis hat es mit dem Steuercode 27, auch ESC genannt. Da für die vielfältigen Möglichkeiten heutiger Drucker 256 Befehle nicht ausreichen, wertet der ESC-Befehl einige ihm folgende Zeichen als Parameter und ermöglicht so die Auswahl der unterschiedlichsten Funktionen. Was die Bedeutung dieser Steuersequenzen anbelangt, so hat hier eine weitestgehende Vereinheitlichung stattgefunden. So wird beispielsweise die Bytefolge 27,83,0, üblicherweise notiert als ESC „S“ (0), wie folgt interpretiert: ESC veranlaßt den Drucker, das folgende Zeichen — auch wenn es, wie in unserem Fall das „S“, druckbar ist — nicht auszugeben, sondern als Befehl zu deuten. „S“ steht konkret für eine Umschaltung auf Hoch- oder Tiefstellen. Die Entscheidung für eine der beiden Möglichkeiten ist vom folgenden Byte abhängig, das wiederum nicht gedruckt, sondern als Parameter gewertet wird. ▶



In unserem Fall bewirkt der Wert 0 ein Hochstellen der Schrift. Die nun folgenden Datenbytes werden wieder ganz normal interpretiert.

Bitte ein Bit: die serielle Schnittstelle

All diese Daten werden von der seriellen Schnittstelle über eine einzige Leitung (TXD = Transmit Data) zum Drucker übertragen. Da diese Leitung vereinbarungsgemäß nur zwei verschiedene Spannungszustände annehmen kann, läßt sich immer nur ein Bit gleichzeitig übermitteln. Um einen Wert von 1 zu senden, legt Amiga diese Leitung auf negatives, für einen Wert von 0 hingegen auf positives Potential. Nach einer durch die Übertragungsrate festgelegten Zeit wird ein dem nächsten Bit entsprechender Spannungswert angelegt, und so geht es Bit für Bit weiter. Eine zweite Leitung (RXD = Receive Data) überträgt auf dieselbe Weise Daten vom Peripheriegerät zum Computer. Damit diese Art der Übertragung funktioniert, müssen die korrespondierenden Schnittstellen auf dieselbe Übertragungsrate eingestellt sein, die in Baud (Bits/Sekunde) gemessen wird. Die weiteren RS232-Anschlüsse führen Versorgungsspannungen oder signalisieren die Betriebszustände der Schnittstellen. GND (Ground) ist die Masseleitung, während CL+ (Current Loop +) und CL- mit +12 beziehungsweise -12 Volt belegt sind. DTR (Data Terminal Ready) und DSR (Data Set Ready) signalisieren die Betriebsbereitschaft der amigaseitigen, peripheren Schnittstelle. Über RTS (Request to Send) und CTS (Clear to Send) tauschen beide Geräte Informationen über Sendeabsicht oder Empfangsbereitschaft. Der serielle Port des Amiga führt noch weitere Signale, die jedoch ohne Bedeutung für den Druckerbetrieb sind.

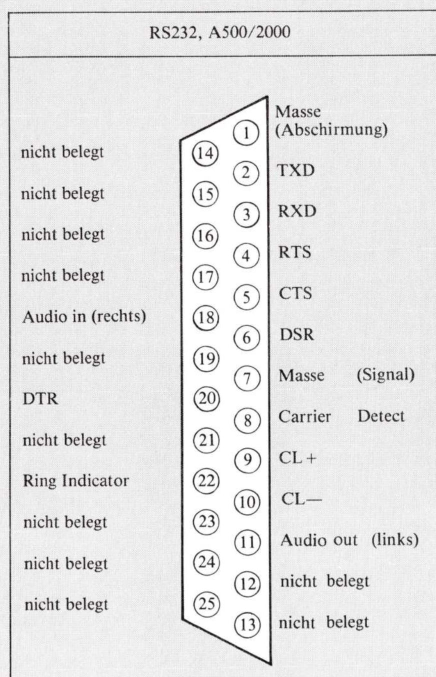
Im Gegensatz zur seriellen Schnittstelle überträgt die Centronics-Schnittstelle jeweils ein komplettes Datenbyte, wobei die Werte aller acht Bits gleichzeitig an acht parallele Datenleitungen gelegt werden, die von 0 bis 7 durchnumeriert sind.

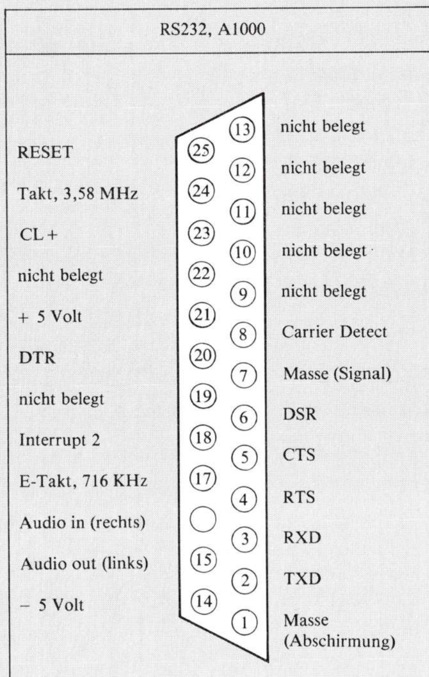
Ein Bit kommt niemals allein: die Centronics-Schnittstelle

Sobald die entsprechenden Spannungswerte bereitstehen, wird dies dem Drucker dadurch mitgeteilt, daß die Strobe-Leitung für eine definierte Zeitspanne auf 0 gesetzt wird. Die geglückte Datenübernahme signalisiert der Drucker wiederum, indem er die Acknowledge-Leitung auf 0 legt, woraufhin Amiga das nächste Byte auf die Reise schickt. Über drei weitere Leitungen, Busy, Paper Out und Select erhält Amiga schließlich Informationen darüber, ob der Drucker empfangsbereit ist, mit Papier versorgt, und ob er sich in On-Line-Stellung befindet. Eine Reset-Leitung sorgt schließlich dafür, daß bei einem Rücksetzen des Amiga auch der Drucker neu initialisiert wird.

Pin-Up: die Anschlußbelegung

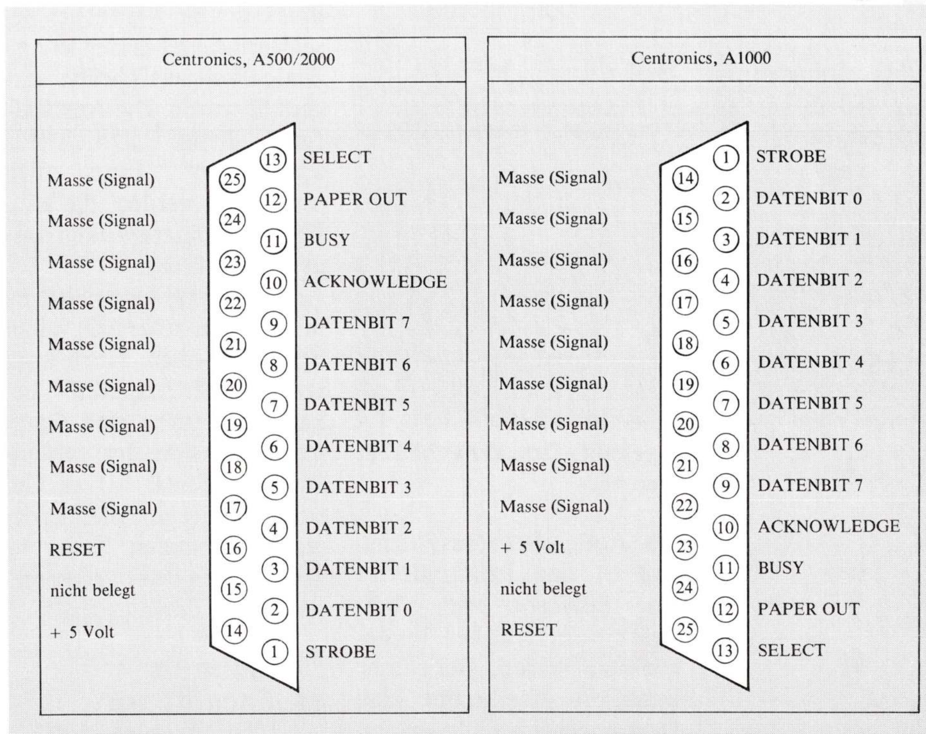
Die Pinbelegung beider Amiga-Schnittstellen wird aus den Abbildungen deutlich. Anschlüsse, die nichts mit der Druckersteuerung zu tun haben, sind mit Kleinbuchstaben beschriftet. Es





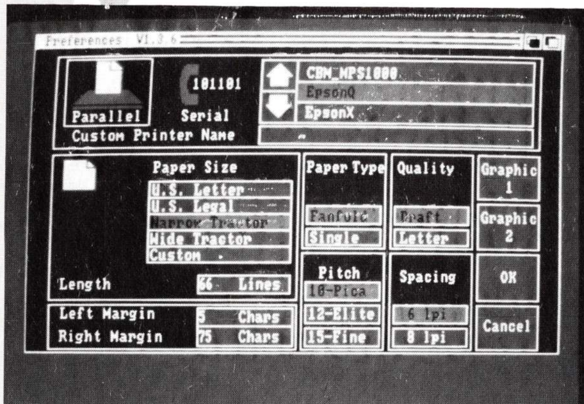
fällt sofort ins Auge, daß Amiga 1000 aus der Reihe tanzt. Bei Amiga 500 und 2000 sind die Ports standardgemäß ausgeführt, hier gibt es hinsichtlich der Verbindungskabel keine Probleme. Bei Amiga 1000 hingegen wird die Centronics-Schnittstelle in unüblicher Weise über einen Stecker nach außen geführt, die RS232 über eine Buchse.

Zudem sind einige Pins abweichend belegt. Besondere Gefahr droht von Pin 23 der Centronics-Schnittstelle, der mit +5 Volt verbunden ist. Wird er mit Pin 23 einer anderen Centronics-Schnittstelle verbunden, dann kommt es unweigerlich zu einem Kurzschluß, der eine teuere Reparatur des Amiga zur Folge haben kann. Amiga 1000-Besitzer müssen beim Kauf eines Druckerkabels besonders auf diesen Gesichtspunkt achten.



Offengeblieben ist bislang die Frage, wie die Schnittstellen von der Software aus mit Daten versorgt werden. Wer Amiga schon näher kennt, weiß, daß das Betriebssystem bestimmte Programme zur Verfügung stellt, die als Verbindungsglieder zwischen (Anwender-)Software und Hardware fungieren und als Devices bezeichnet werden. Für die Druckeransteuerung kommen theoretisch gleich drei verschiedene Devices in Betracht: das Parallel.device, das Serial.device und das Printer.device; diese entsprechen den AmigaDOS-Geräten par:, ser: und prt:. Alle drei Devices befinden sich im Unterverzeichnis „devs“ der Workbench-Diskette. Wie schon der Name verrät, ist das Printer.device d a s Gerät, über welches die Druckersteuerung ablaufen sollte. Da par: und ser: die jeweilige Schnittstelle direkt ansprechen, muß ein Programm, das Daten über diese Geräte ausgibt, die drukkereigenen Steuersequenzen verwenden. Sobald man einen Drucker mit anderem Befehlssatz anschließt, müssen dann die entsprechenden Programmabschnitte dahingehend abgeändert werden. Das Printer.device ist in dieser Hinsicht wesentlich flexibler. Es kennt einen eigenen Befehlssatz, über den es von allen Programmen einheitlich angesprochen werden kann. Dieser Befehlssatz umfaßt mehr als 70 unterschiedliche Steueranweisungen, die vom Printer.device in eine für den jeweils vorgesehenen Drucker verständliche Form umgeformt werden. Hierzu muß das Printer.device die Charakteristika des angeschlossenen Druckers kennen. Diese werden auf einfache Weise im Workbench-Programm „Preferences“ festgelegt, wo sie vom Printer.device abgefragt werden. Die wichtigsten Einstellungen, die hier zu tätigen sind, betreffen die Wahl des passenden Druckertreibers und der Schnittstelle, an die der Drucker angeschlossen ist. Über den Treiber erhält das Printer.device alle nötigen Informationen, es kann nun auch die „Printer.device-Steuersequenzen“ in die druckerspezifischen umwandeln und diese dann über die Druckertreiber an die gewünschte Schnittstelle weitergeben. Der Vorteil dieses Verfahrens liegt auf der Hand. Mit vergleichsweise wenig Aufwand lassen sich Programme entwickeln, die mit allen Druckern zusammenarbeiten, zu denen ein passender Treiber existiert, vorausgesetzt natürlich, daß die Druckerfähigkeiten den Anforderungen des Programms gerecht werden können.

Grundsätzlich positiv anzurechnen ist den Entwicklern ▶



Wie gehabt: Druckerwahl über Preferences

des Amiga-Betriebssystems, daß in das Printer.device eine Funktion zum Ausdruck von Grafiken implementiert wurde. Bei dieser Funktion übernimmt das Device zunächst die in den Preferences vorzunehmenden Grundeinstellungen, die unter anderem den angeschlossenen Druckertyp, die Ausrichtung des Bildes auf dem Papier und den Darstellungsmodus (S/W, Graustufen oder Farbe) betreffen. Für den Ausdruck werden dann die Daten der Bitplanes, in denen die Grafikinformationen von Amiga abgelegt sind, in Werte umgerechnet, die den Drucker zum Setzen der entsprechenden Punkte veranlassen. Leider geht diese Umrechnung quälend langsam vonstatten. Während sonst der Computer auf den Drucker warten muß, ist es hier umgekehrt. Die Zeit scheint schier stillzustehen, wenn der Druckkopf zwischen den einzelnen Zeilen auf die nächsten Daten wartet. Die Ursache hierfür liegt in den Routinen des Printer.device, die einfach zu umständlich arbeiten. Daß es wesentlich schneller gehen kann, zeigt ein von Irseesoft entwickeltes Programm namens TurboPrint, das den Grafikdruck um ein Vielfaches beschleunigt und in der letzten AmigaWelt ausführlich vorgestellt wurde. Die Anschaffung dieses „Druckerbeschleunigers" kann allen empfohlen werden, die häufiger Computergrafiken ausdrucken müssen.

Auch Commodore selbst ist in der Zwischenzeit nicht untätig gewesen. Noch kurz vor Redaktionsschluß bekam der Autor eine Gamma-Testversion der geplanten Workbench 1.3 zu Gesicht, die neben vielen anderen interessanten Neuerungen auch ein stark verbessertes Printer.device enthält.

Mehr als Facelifting: Workbench 1.3

Es verspricht einen 3 bis 20 mal schnelleren Grafikdruck, abhängig von verschiedenen Faktoren wie Druckertyp, Darstellungsmodus, Ausrichtung des Bildes, Verhältnis freibleibender zu der zu bedruckenden Fläche etc. Eine schnelle Probe aufs Exempel wurde mit einem NEC P6 gemacht. Hierzu wurde mit beiden Devices die Hardcopy eines PAL-Bildschirmes erstellt. Als Voreinstellungen wurden Schwarz-Weiß-Druck, Treshold = 8 sowie horizontale Ausrichtung gewählt. Mit dem bisherigen Device nahm diese Prozedur satte 22,12 Minuten in Anspruch, mit dem neuen Device verkürzte sich die Wartezeit auf 2,29 Minuten — Druckköpfe werden demnächst auf ihre gewohnten Erholungspausen verzichten müssen. Neben schnelleren Druckroutinen verfügt das Printer.device 1.3 auch über die schmerzlich vermißte Funktion zum Ausdrucken von HAM-Bildern, die übrigens auch bei TurboPrint implementiert ist. Offiziell wird die Workbench 1.3 in Deutschland nie erscheinen. Da sie aber in USA vertrieben wird, wird Commodore Deutschland die Dokumentation dieser Version vornehmen. Über die üblichen PD-Kanäle wird jedoch jedermann bald zu seiner neuen Arbeitsplatte kommen. Der Grund für Commodores Verhalten ist einleuchtend: im Herbst steht die Workbench 1.4 auf dem Lieferprogramm — wozu da noch 1.3 veröffentlichen?

Die Wahl des richtigen Treibers ist die entscheidende Hürde, die es bei der Druckeranpassung zu überwinden gilt. In Tabelle 1 finden Sie eine Übersicht über die Amiga-Druckertreiber. Diese Treiber sollten sich im Unterverzeichnis „devs/printers" der Workbench oder im gleichnamigen Verzeichnis der Extras-Diskette befinden. Die Auslagerung einiger Treiber hat einen einfachen Grund in dem Platzmangel, der auf der aus allen Nähten platzenden Workbench herrscht. Sollte sich der benötigte Druckertreiber auf der Extras-Diskette befinden, muß er zunächst in das devs/printers-Verzeichnis der Workbench kopiert werden. Da die Treiber keine Icons besitzen, muß dies vom CLI aus geschehen. Wer noch keine CLI-Erfahrung besitzt, sollte das AmigaDOS-Handbuch oder ein entsprechendes Lehrbuch zu Rate ziehen. Da sich die Ausstattung der Workbench-Disketten mehrmals geändert hat, kann es auch passieren, daß Sie einen Treiber vergeblich suchen. In solchen Fällen können Sie einen Händler oder einen Amiga-User-Club um Unterstützung bit-

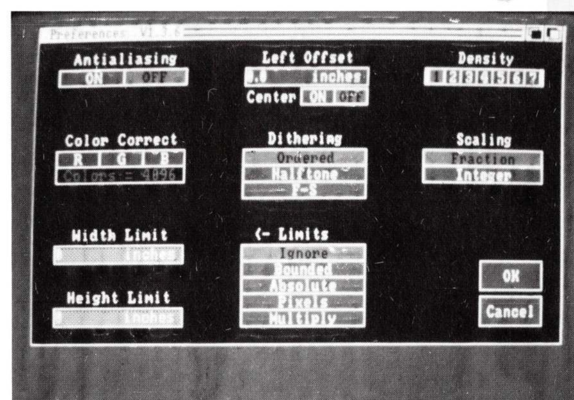
ten. Bei einem sehr seltenen Drucker ist möglicherweise ein passender Treiber direkt vom Hersteller zu erhalten. Die geringsten Probleme macht die Anpassung natürlich dann, wenn sich ein Treiber findet, dessen Bezeichnung mit der des Druckers übereinstimmt. Findet sich ein solcher nicht, dann empfiehlt es sich, das Druckerhandbuch zu Hilfe zu nehmen. Die meisten Drucker haben nämlich nicht nur ihren eigenen Befehlssatz, sie sind auch in der Lage, andere Druckertypen zu emulieren, das heißt: sie verhalten sich so, wie der emulierte Drucker es tun würde. In dieser Betriebsart läßt sich der Drucker dann über den Treiber des emulierten Typs ansteuern. Die höchste „Trefferquote" dürften hier die Epson-Treiber erzielen, da Epson-Befehlssätze eine Quasi-Norm darstellen, die eine Emulation am lohnenswertesten erscheinen läßt. Es kann auch vorkommen, daß baugleiche Drucker mit unterschiedlichen Namen auf dem Markt sind. Bekanntestes Beispiel ist der NEC P6, baugleich mit dem MPS 2000 von Commodore, der dann auch den CBM_MPS2xxx-Treiber benötigt. Wenn all diese Wege nicht zum Ziel führen, hilft gegebenenfalls das Durchprobieren der einzelnen Treiber oder — besser noch — eine Anfrage beim Händler oder Druckerhersteller. Sollten gar alle Stricke reißen, was wohl nur in extremen Ausnahmefällen vorkommen wird, dann muß man in den sauren Apfel beißen und den Treiber selbst erstellen. Da eine direkte Programmierung eines solchen Treibers detaillierteste Systemkenntnisse erfordert, greift man besser auf die Public Domain zurück. Auf Disk 90 von Fred Fish befindet sich das Programm PrtDrvGen, das im Dialog mit dem Benutzer alle erforderlichen Parameter, die dem Druckerhandbuch zu entnehmen sind, erfragt und aus ihnen den passenden Treiber generiert.

Mal hier — mal da: Drucker an der PC-Karte

Amiga 2000-Besitzer mit einer PC-Karte können einen an der Amiga-Centronics-Schnittstelle angeschlossenen Drucker auf einfache Weise auch vom PC aus nutzen. Hierzu muß das Programm „LPT1" als Hintergrundtask laufen. Nun kann der PC die Schnittstelle ansprechen, während sie für den Amiga-Zugriff blockiert ist. Für die Druckeranpassung ist dann allerdings die jeweils installierte MS DOS-Software zuständig.

Partnerwahl 2: User sucht Drucker

Wie gesehen, lassen sich fast alle Drucker am Amiga betreiben. Dieser Umstand macht die Kaufentscheidung nicht gerade einfacher. Die wichtigsten Gesichtspunkte, die vor dem Kauf beachtet werden sollten, haben wir für Sie in einer Tabelle zusammengestellt. Auch bei Amiga dürften — trotz seiner Grafikqualitäten — die meisten Anwender den Einsatzschwerpunkt eines Druckers im Textverarbeitungsbereich ansiedeln. Wer Texte in echter Korrespondenzqualität zu Papier bringen möchte, auf Grafikdruck aber völlig verzichten, kann zu Typenraddruckern greifen, die ein exzellentes Schriftbild aufweisen, bauartbedingt jedoch sehr laut und langsam sind. Sehr hohe Schriftqualität bei quasi unbegrenzter Anzahl von Schriftarten bieten die Laserprinter, die demnächst in der AmigaWelt ausführlich besprochen werden. Wer in professionellem Stil Satzarbeiten oder Desktop-Publishing betreiben möchte, kommt an diesen Druckern nicht vorbei. Hochwertige 18- und 24-Nadel-Drucker erreichen fast Korrespondenzqualität, der Unterschied ist mit bloßem Auge oft nur sehr schwer auszumachen. Im Vergleich zu den Typenraddruckern sind sie — auch im Schönschrift-Modus — wesentlich schneller, allerdings auch fast genau so laut. Die Einzelansteuerung der Nadeln läßt sowohl Grafikdruck als auch verschiedene Schriftsätze zu. Bei einigen Modellen lassen sich Zeichensätze in Form von Steckmodulen nachrüsten, bei anderen können frei definierbare Zeichensätze aus dem Computer nachgeladen werden. Diese sehr flexible Möglich- ▶



Zweites Auswahlmenü der 1.3 Preferences für Grafikfeineinstellung

keit wird leider nicht allzu oft von der Software ausgenutzt. Ebenfalls vom Modell abhängig ist die Möglichkeit, Texte farbig zu gestalten. Zusammenfassend lassen sich die 18- und 24-Nadler als die Allzweckdrucker überhaupt für Amiga bezeichnen.

Bedingt trifft diese Aussage auch noch für die 9-Nadler zu. Sie sind so vielseitig wie ihre großen Brüder, können aber hinsichtlich Auflösung, Schriftqualität und Druckgeschwindigkeit nicht mit ihnen mithalten. Für sie spricht der günstige Preis, zu dem sie mittlerweile erhältlich sind.

Störend auswirken kann sich die Lautstärke dieser sogenannten Impact-Drukker dann, wenn sie an Standplätzen eingesetzt werden, an denen mehrere Personen gleichzeitig arbeiten. Obwohl spezielle Schallschutzhauben Abhilfe schaffen können, sind Tintenstrahl- und Laserdrucker in diesen Fällen die bessere Alternative.

Für Grafikdruck können Heimanwender auf die Matrixdrucker zurückgreifen, die fast alle grafikfähig sind. Farbdruck ist jedoch nur bei wenigen Modellen möglich. Bilder mit hoher Auflösung bringen jedoch sehr lange Druckzeiten mit sich. Die Leuchtkraft der Farben reicht nicht an hochwertige Farbdrucker heran und das Farbband ist nach nur wenigen Ausdrucken so verschmiert, daß es keine optimalen Ergebnisse mehr liefern kann. Für professionelle Anwendungen kommen daher nur Tintenstrahl-, Thermotransfer- oder Laserducker in Betracht, wobei letztere allerdings nicht farbfähig sind. Nicht jeder gute Grafikdrucker verbindet mit dieser Fähigkeit eine gleichfalls gute Schriftqualität. So gibt es beispielsweise Tintenstrahldrucker, die nicht einmal Unterlängen drucken können. Wer sowohl exzellenten Farbdruck als auch hohe Schriftqualität braucht, kommt möglicherweise um die Anschaffung eines zweiten Druckers nicht herum.

Von nicht zu unterschätzender Bedeutung, gerade im professionellen, aber auch im Homeanwenderbereich, ist der Bedienungskomfort. Ein ausführliches, leicht verständliches — möglichst auch deutschsprachiges — Handbuch sollte eine Selbstverständlichkeit sein. Alle Bedienelemente müssen übersichtlich angeordnet und leicht erreichbar sein, ein Farbband- oder Kartuschenwechsel muß schnell und ohne schmutzige Finger vonstatten gehen. Dem Papierhandling ist — den eigenen Ansprüchen entsprechend — besondere Aufmerksamkeit zu widmen. Ein einfacher Traktor für Endlospapier oder ein halbautomatischer Einzelblatteinzug können für den Hausgebrauch durchaus ausreichen, für Büroanwendungen hingegen sind bidirektionale Traktoren, die das Papier vorwärts und rückwärts transportieren können, und ein vollautomatischer Einzelblatteinzug unabdingbar. Muß mit mehreren Formularen gleichzeitig gearbeitet werden, wird sogar ein Doppel- oder Mehrschachteinzug erforderlich. Sehr praktisch sind Drucker, die einen Wechsel zwischen Endlos- und Einzelblattpapier gestatten, ohne daß hierzu der Drucker umgebaut werden muß.

Von Bedeutung sind ebenfalls das Papierformat, das der Drucker verarbeiten kann, und die Größe des Zeichenpuffers. In diesem werden die vom Computer kommenden Daten zwischengespeichert (1 DIN-A4-Seite entspricht zirka 6 KByte), so daß der Drucker diese Informationen abarbeiten kann, während der Rechner schon wieder für andere Aufgaben zur Verfügung steht. Solch ein Puffer ist auch bei einem Multitaskinggerät wie Amiga von Bedeutung, da viele Programme diese Fähigkeit nicht voll ausnutzen und der Programmablauf vom Druckvorgang unterbrochen wird. Für einige Anwendungen mag die Fähigkeit, Durchschläge zu erstellen, wichtig sein. Hier kommen nur Matrix- und Typenrad-

Bezeichnung des Treibers	Druckertyp
Alphacom_AlphaPro_101	Typenrad
Brother_HR-15XL	Typenrad
CanonLBP-8	Laser
CBM_MPS1000	Nadel
CBM_MPS1500	Nadel
CBM_MPS2xxx	Nadel
Diablo_630	Typenrad
Diablo_Advantage-D25	Typenrad
Diablo_C-150	Tinte
Epson	Nadel
EPSON_EX-800,1000	Nadel
Epson-JX-80	Nadel
EPSON_LQ-800,1000	Nadel
FUJITSU_DX2000	Nadel
HP_Laserjet	Tinte
HP_Laserjet_PLUS	Tinte
ImagewriterII	Nadel
Okidata_292	Nadel
Okidata_92	Nadel
Okimate_20	Thermotransfer
Qume_LetterPro_20	Typenrad
Xerox_4020	Tinte
Okidata_293I*	Nadel
Epson X*	Nadel
Epson Q*	Nadel
HP_PaintJet*	Tinte

*ab Workbench 1.3

Tabelle 1: Amiga Druckertreiber

drucker in Betracht. Auch nach den Reparaturbedingungen sollte man sich erkundigen. Was nutzt der beste Drucker, wenn er bei einem Defekt mehrere Monate auf Reisen ist! In die finanziellen Überlegungen sind auch die Betriebskosten einzubeziehen. Sie werden in erster Linie von den Papierkosten (Spezialpapier?) und den Kosten für Farbbänder/-kartuschen und deren Lebensdauer bestimmt.

Eigeninitiative gefragt!

Mit diesem Hintergrundwissen ausgestattet, sollten Sie nun in der Lage sein, Ihre persönlichen Anforderungen hinsichtlich des Druckers zu konkretisieren. Haben Sie erste Vorstellungen von Ihrem Drucker, dann durchforsten Sie am besten Computerzeitschriften wie die AmigaWelt. Hier finden Sie regelmäßig Testberichte zu den aktuellen Modellen. Studieren Sie auch die Anzeigenseiten. Suchen Sie dann die Händler auf, und lassen Sie sich die Geräte, die für Sie in Frage kommen, vorführen, am besten gleich am Amiga. Ich wünsche Ihnen, daß Sie eines dieser Geschäfte zufrieden verlassen, den passenden Drucker unter dem Arm.

(Ralf Leithaus)

Wichtige Gesichtspunkte beim Druckerkauf

1.Textausdruck
a) Schriftqualität
b) Druckgeschwindigkeit
c) verfügbare Zeichensätze
d) Farbdruck

2.Grafikdruck
a) Bildqualität
b) Druckgeschwindigkeit
c) Farbdruck

3.Handhabung
a) verständliche Dokumentation
b) übersichtliche Bedienfelder
c) praktisches Papierhandling
d) bequemer Farbbandwechsel

4.Verschiedenes
a) maximale Papiergröße
b) Puffergröße
c) Drucken mit Durchschlägen
d) Reparatur- und Wartungsservice

5.Kosten
a) Anschaffungskosten
b) Betriebs- und Wartungskosten

Nadel

Im Druckkopf eines Nadeldruckers befinden sich Nadeln mit punktförmiger Spitze, die von einer elektromagnetischen Spule gegen das Farbband gedrückt werden und so Punkte auf das Papier setzen. Die verschiedenen Modelle weisen eine unterschiedliche Anzahl von Nadeln auf (von 8 bis 48), die in Matrixform angeordnet sind. Je nach Anzahl der Nadeln läßt sich die Zusammensetzung der Zeichen aus Einzelpunkten mehr oder weniger deutlich erkennen. Beim Drucken erzeugen die Nadeln ein lautes, recht unangenehmes Geräusch.

Tintenstrahl

Tintenstrahldrucker arbeiten ähnlich wie Nadeldrukker, jedoch sind statt Nadeln enge Düsen im Druckkopf untergebracht, die die Tinte in kleinen Tröpfchen auf das Papier sprühen. Sie arbeiten nahezu geräuschlos.

Laser

Laserdrucker arbeiten vom Prinzip her wie Fotokopiergeräte. Eine Trommel wird an den zu bedruckenden Stellen statisch aufgeladen und zieht an diesen Stellen die Tonerpartikel an, die anschließend auf das Papier gewalzt werden. Die Aufladung der Trommel geschieht hier allerdings nicht über eine Belichtungsoptik, sondern einen gesteuerten Laserstrahl. Laserdrucker sind sehr leise und schnell und haben eine hohe Auflösung, sie können jedoch nur schwarz-weiß drucken.

Thermotransfer

Thermotransferdrucker verwenden besondere Farbbänder mit wärmeempfindlicher Tinte. Diese Tinte wird von den Heizelementen des Druckkopfes erwärmt und regelrecht auf das Papier geschweißt. Ein Verlaufen der Tinte ist bei diesem — ebenfalls nahezu lautlosen — Verfahren nicht möglich.

Typenrad

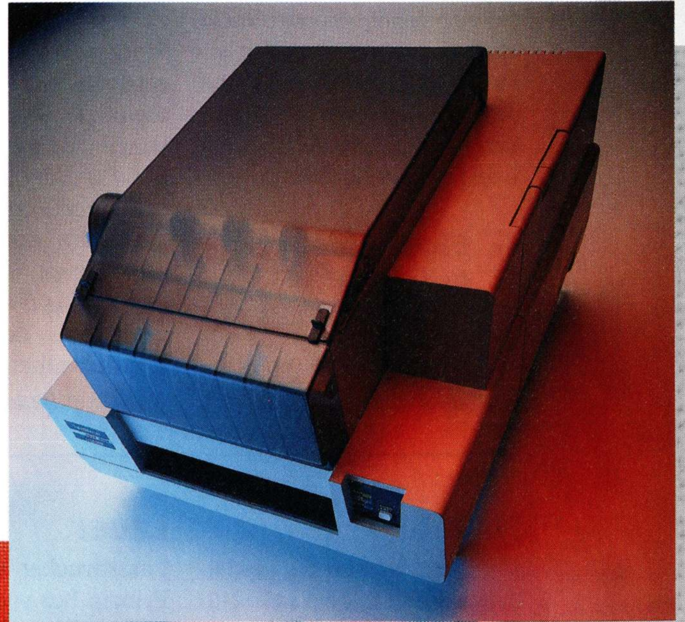
Bei einem Typenraddrucker sind die zu druckenden Typen auf dem Rand einer Scheibe angeordnet. Zum Drucken werden diese Typen durch Drehen des Rades vor einen elektromagnetisch gesteuerten Stift positioniert, der sie gegen dasFarbband drückt — ein sehr lautes und langsames Verfahren. Der Zeichensatz läßt sich nur durch Wechseln des Typenrades ändern. Die Schrift erreicht dafür Korrespondenzqualität.

Plasma

Bei diesem neuartigen Druckverfahren, das sich noch in der Entwicklung befindet, wird der Farbstoff in kleine Teilchen zerstäubt, die elektrostatisch aufgeladen werden. Diese Partikel werden zu einem Strahl gebündelt, der von Kondensatorplatten abgelenkt und so auf die korrekte Druckposition ausgerichtet wird.

Meister seines Fachs: CalComp PaintMaster

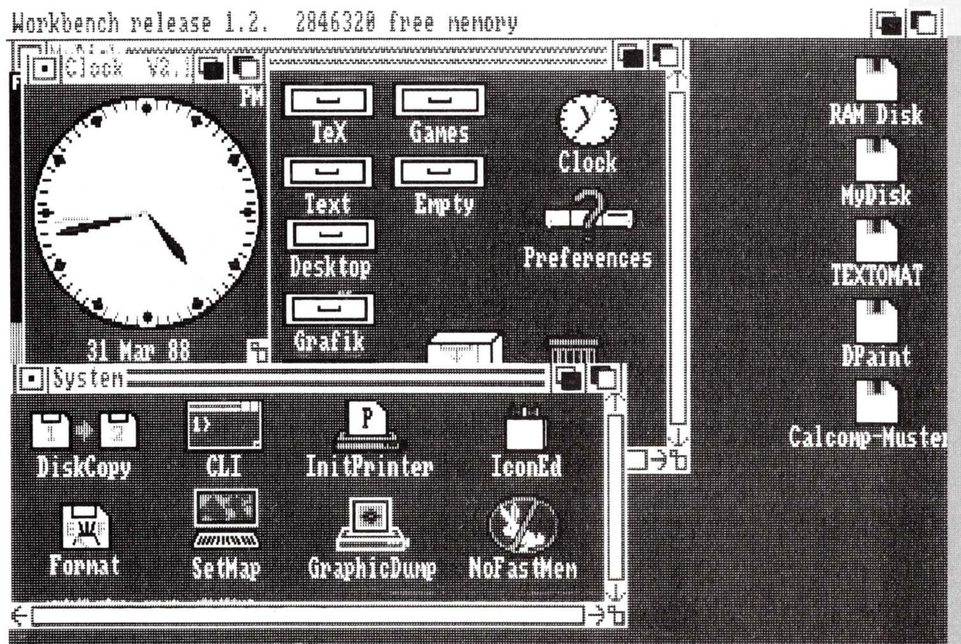
Die Auswahl an hochwertigen „Farbdrukkern", die die Grafikfähigkeiten vom Amiga in adäquater Weise zu Papier bringen können, ist nicht groß. Mit dem PaintMaster der Firma CalComp stellen wir einen Thermotransferdrucker vor, der speziell für unseren Rechner konzipiert ist und höchsten Ansprüchen gerecht wird.





Ausdruck des bekannten Digi-View-Demos.

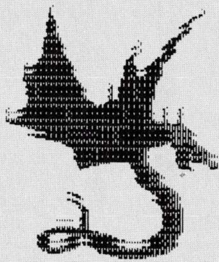
Professionelle Grafik- und CAD-Anwender benötigen Drucker, die Farbgrafiken schnell und in hochwertiger Qualität zu Papier bringen. Mit dem Mehrfarbenplotter CalComp PaintMaster ist ein für diesen Anwenderkreis geeignetes Gerät verfügbar. Da PaintMaster auch Folien bedruckt, eignet er sich auch hervorragend zur Erstellung von Vorlagen zu Unterrichtszwecken und für professionelle Präsentationen.
PaintMaster arbeitet nach dem Thermotransferverfahren und erreicht eine Auflösung von 203,2 Punkten/Zoll horizontal bzw. 200 Punkten/Zoll vertikal. Dabei kann PaintMaster die Farben gelb, magenta, zyan, rot, grün, blau und schwarz sowie in verschiedenen Zwischentönen darstellen.
Der Anschluß des Gerätes an den Amiga ist unproblematisch, da hierfür sowohl eine Centronics- als auch eine RS232-Schnittstelle zur Verfügung stehen. Mit den wenigen Bedienelementen wird man dank eines verständlichen und ausführlich bebilderten deutschsprachigen Handbuchs schnell vertraut.
An der Vorderseite von Paintmaster befindet sich neben 4 Leuchtdioden, die über Betriebszustand bzw. Fehlerstatus informieren, nur eine EJECT-Taste, die zum Abbrechen eines Ausdrucks bei gleichzeitiger Leerung des Druckpuffers dient. An der rechten Geräteseite entdeckt man neben dem Netzschalter einen Netzspannungswähler, einen Sicherungshalter und den Einschub für die Farbbandkassetten. An der Rückseite finden sich neben den Schnittstellenports zwei gut zugängliche DIP-Schaltergruppen, die für die Parameter der seriellen Datenübertragung und die Wahl des nationalen Zeichensatzes zuständig sind.
So informiert, läßt sich der erste Ausdruck in Angriff nehmen. Die Farbbandkassette ist rasch eingesetzt, auch der im Lieferumfang enthaltene automatische Einzelblatteinzug ist schnell gefüttert. Er sitzt auf dem Gerät und kann 100 Blatt Papier bzw. 40 Folien aufnehmen. Hier ist die Verwendung von CalComp-Papier und -farbbändern zu empfehlen, da diese Komponenten optimal auf PaintMaster abgestimmt sein sollten, um beste Druckqualität zu garantieren. Die Spannung wächst. Amiga ein, Drucker ein, Workbench geladen, in Preferences den Calcomp-Druckertreiber eingestellt, DigiPaint gestartet, schnell ein Bild geladen. Jetzt kann die Print-Option angewählt werden. Der Autor neigt nicht gerade zu Mystifikationen, aber der nun folgende Druckvorgang hat schon etwas Geheimnisvolles an sich. Nach einigen Sekunden wird ein Blatt Papier eingezogen, und nur das Blinken der Wait-Anzeige ▶



Kräftige und klare Farbumgebung auch in Schwarzweiß.

CALCOMP PAINTMASTER

Normal:	ABCDEFGHIJKLMNOPQRSTUVWXYZ ÄÖÜ ß 0123456789
Fett:	**ABCDEFGHIJKLMNOPQRSTUVWXYZ ÄÖÜ ß 0123456789**
Kursiv:	*ABCDEFGHIJKLMNOPQRSTUVWXYZ ÄÖÜ ß 0123456789*
Unterstrichen:	ABCDEFGHIJKLMNOPQRSTUVWXYZ ÄÖÜ ß 0123456789
Hochsetzen:	ABCDEFGHIJKLMNOPQRSTUVWXYZ ÄÖÜ ß 0123456789
Tiefsetzen:	ABCDEFGHIJKLMNOPQRSTUVWXYZ ÄÖÜ ß 0123456789
Elite:	ABCDEFGHIJKLMNOPQRSTUVWXYZ ÄÖÜ ß 0123456789
Pica:	ABCDEFGHIJKLMNOPQRSTUVWXYZ ÄÖÜ ß 0123456789
Schmal-Schrift:	ABCDEFGHIJKLMNOPQRSTUVWXYZ ÄÖÜ ß 0123456789
Breit-Schrift:	ABCDEFGHIJKLMNOPQRST



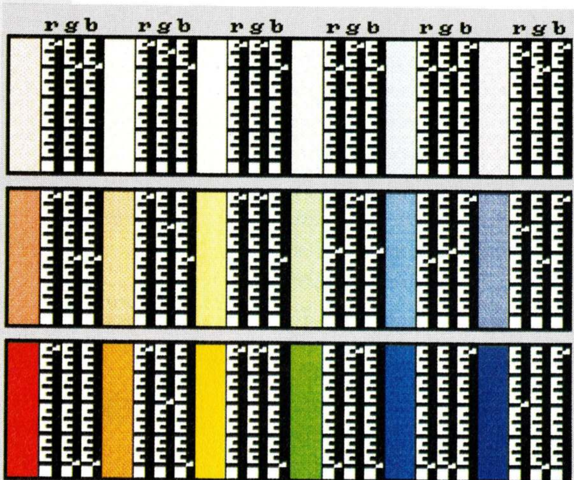
Schönschriften in CalComp-Qualität

kündet davon, daß PaintMaster die Arbeit aufgenommen hat. Von Zeit zu Zeit verrät ein leises Surren den Auftrag der nächsten Farbschicht. Nach zirka vier Minuten wird das fertige Bild ausgegeben. Das Ergebnis ist beeindruckend. Es besticht besonders durch die Leuchtkraft der Farben und die absolute Gleichmäßigkeit des Farbauftrags. Schlieren, wie man sie von Matrixdruckern her kennt, sind für PaintMaster kein Thema. Eine hochgenaue Papierführung sorgt schließlich dafür, daß keine Freiräume zwischen den Zeilen entstehen.

Auch die Grafiken in Schwarzweiß- bzw. Graustufendruck überzeugen. Sie lassen sich hervorragend zur einfachen und flexiblen Herstellung einfarbiger Druckvorlagen, z. B. für Handbücher und Programmbeschreibungen, verwenden.

Obwohl der Einsatzschwerpunkt von PaintMaster eindeutig im Grafikbereich liegt, kann sich seine Schriftqualität im Text-Modus — ein Schwachpunkt vieler Grafikdrucker — sehen lassen. In dieser Betriebsart emuliert PaintMaster den 9-Nadel-Drukker FX-80 von Epson, er muß hierzu vom Epson-Druckertreiber der Workbench angesprochen werden. Wie man dem Beispielausdruck entnehmen kann, ist die NLQ-Schrift sehr schön, hinsichtlich des Kontrastes und der wiederum absolut gleichmäßigen Farbsättigung wird das Vorbild noch übertroffen. Bei der Kursivschrift muß man einige Abstriche machen, sie zeigt eine für 9-Nadel-Drukker typische Treppenstruktur. Mit entsprechenden Programmen ist auch die Einbindung von Schwarzweiß-Grafiken in den Text kein Problem. Der Ausdruck einer Textseite dauert nur 45 Sekunden, so daß PaintMaster auch für weite Bereiche der Textverarbeitung eingesetzt werden kann. Professionellen Ansprüchen kann er in diesem Aufgabenfeld allerdings nicht gerecht werden.

Das Arbeiten mit PaintMaster gestaltet sich sehr angenehm und unkompliziert, wozu im wesentlichen die hohe Arbeitsgeschwindigkeit des Druckers beiträgt.



Farbtestpalette als Vergleichsgrundlage: CalComp-Ausdruck

Großer Puffer verschafft dem Computer Spielraum

Ein Pufferspeicher von 1,5 MByte sorgt zudem dafür, daß der Rechner schnell wieder für andere Aufgaben bereit steht. Ist ein Farbband verbraucht, so wird dies über dieKontroll-LED's angezeigt, und PaintMaster unterbricht den Betrieb, bis ein neues Farbband eingelegt ist, was mühelos zu bewältigen ist. Nicht zuletzt sorgt auch die flüsterleise Lautstärke für ein angenehmes Arbeiten mit PaintMaster.

Angesichts dieser Leistungsmerkmale kann der derzeitige Listenpreis von 11 700 Mark + 14% MwSt. als angemessen bezeichnet werden. Die obere Grenze ist

hiermit allerdings noch nicht erreicht. Die CalComp-Palette umfaßt auch noch die Geräte ColorMaster und PlotMaster, die in erster Linie für den Grafikeinsatz am PC/AT gedacht sind. Sie besitzen gegenüber PaintMaster zusätzliche Fähigkeiten, wie größere Puffer, eingebauten Rasterizer und die Verarbeitung von Objektgrafikdaten. Sie dürften für Amiga 2000-Besitzer mit PC-Karte und der entsprechenden MS-DOS-Software interessant sein. Auf der CeBIT wurde kürzlich die Produktreihe 12 vorgeführt, die eine Verarbeitung der großen DIN-A3-Formate möglich macht.

Druckertreiber erst auf Workbench 1.3

Leider wird das ansonsten durchweg positive Bild von PaintMaster durch völlig überflüssige Probleme bei der Beschaffung des Druckertreibers getrübt. Der Treiber selbst fehlt auf den standardmäßigen Workbench- und Extras-Disketten, ist aber inzwischen von Commodore entwickelt worden und wird von CalComp mit dem Gerät ausgeliefert.

(Ralf Leithaus)

Info: CalComp GmbH, Hansa Allee 197, 4000 Düsseldorf 11, Tel.: 02 11/ 5 20 80
Preise: Drucker 11 700 Mark + MwSt., Farbbänder 199 Mark + MwSt. (145 Grafikausdrucke), 500 Blatt satiniertes Papier ca. 35 Mark, 100 Folien A4 164 Mark. Kosten pro Grafikausdruck 1,45 (Folie ca. 3 Mark) — alles zuzüglich MwSt.

Technische Daten:

Drucktechnik:	Thermotransfer
Auflösung:	203,2 dpi/8,00 Punkte pro mm horizontal 200,0 dpi/7,87 Punkte pro mm vertikal
Farben:	gelb, magenta, zyan, rot, grün, blau, schwarz, plus Zwischentöne
Druckzeit:	1,45 Minuten bei Farbgrafiken 45 Sekunden schwarzweißer Text
Automatischer Einzelblatteinzug:	100 Blatt Papier 40 Folien
Thermo-Farbband:	145 Bilder Mehrfarbenband 500 Bilder schwarzes Farbband (Typ „single-Pass")
Schnittstellen:	RS-232-C seriell Centronics parallel
Betriebsarten:	Drucker-Modus Rastergrafik-Modus
Plotfläche:	200 mm × 254 mm 8 × 10 Zoll
Zeichenmaterial:	Papier (Einzelblätter) oder Folien
Format:	A — 8,5 × 11 Zoll (216 mm × 279 mm) A4 — 8,24 × 11,69 Zoll (210 mm × 297 mm)
Internes RAM:	1.536 KB
Zeichensätze:	96 ASCII-Zeichen mit Kursivschrift, 32 fremdspachliche Zeichen
Zeichenmatrix:	16 Punkte horizontal × 24 Punkte vertikal
Zeichenabstand:	10, 12 und 16,5 Zeichen pro Zoll 80 Zeichen/Zeile bei 10 cpi 96 Zeichen/Zeile bei 12 cpi 132 Zeichen/Zeile bei 16,5 cpi
Zeilenabstand:	6, 8 und 10 Zeilen pro Zoll
Abmessungen:	Breite: 432 mm Tiefe: 508 mm Höhe: 305 mm
Gewicht:	16,4 kg
Betriebstemp.:	10 Grad C bis 35 Grad C
Lagertemperatur:	-20 Grad C bis 40 Grad C
Rel. Luftfeuchte:	Betrieb: 20 bis 80 Prozent
Lagerung:	20 bis 80 Prozent ohne Kondensierung
Geräuschpegel:	unter 55 dB
Netzanschluß:	einphasig, Sternspannung über Schalter wählbar: 100-120 V, +/- 10% oder 200-240 V, +/- 10 %, 47-63 Hz
Wärmeentwickl.:	258 kcal/h
Sicherung:	1,5 Ampere

Drucken mit MPS 1200P

Für den User mit kleinem Geldbeutel genau der richtige Drucker: MPS1200P kennt wie kein anderer jeden Amiga-Modus. Wie man die Ansteuerung handhabt, erläutert folgender Artikel.



Druckerchamäleon: MPS 1200P paßt sich in seiner (Rechner-)Umgebung an wie kein anderer.

Prinzipiell gelten die folgenden Ausführungen für alle Drukker, die den ESC/P-Code verstehen. Als Beispiel wurde der Commodore-MPS1200P gewählt, weil dieser nicht nur die Amiga-, sondern auch die Epson- und IBM-Betriebsart beherrscht. Normalerweise muß zur Übertragung von Daten an den Drucker keine Datei geöffnet werden. Es genügt dafür der Befehl LPRINT oder zur Ausgabe eines Listings LLIST. Das Ausgabeformat hängt von den Einstellungen in Preferences ab. Für den MPS1200P wählt man den Epson-Druckertreiber. Alle DIP-Schalter bleiben auf OFF. Wer will, kann auch Nummer 4 einschalten. Diese Konfiguration liefert den deutschen Epson-Zeichensatz und die mit Preferences gewählten Parameter (Zeilenabstand, Rand, Schriftart usw.).
Als Alternative kann auch ein Druckerfile geöffnet werden: OPEN „prt:“

```
FOR OUTPUT AS #1
```

Statt LPRINT muß dann PRINT #1 verwendet werden und anstelle von LLIST ist LIST, „prt:“ zu wählen. Dazu muß die Datei geschlossen sein!
Auch die Einstellung des Druckers kann per Programm geändert werden. Als Beispiel die Umschaltung auf Schönschrift (NLQ):

```
OPEN"prt:" FOR
 OUTPUT AS #1
nlq$=CHR$(27)+
 "[2"+CHR$(34)+"z"
PRINT #1,nlq$:
  CLOSE #1
```

Einfacher, ohne eine Datei öffnen zu müssen, geht das mit:

```
LPRINT CHR$(27);
"[2;CHR$(34);"z"
```

Wie ein Blick in die Tabelle der ANSI-Steuercodes zeigt, ist die Umschaltung der Konfigurationen (Amiga, Epson, IBM) mit diesen Codes nicht möglich. Daher kann der komplette Amiga-Zeichensatz mit dem Epson-Druckertreiber nicht ausgedruckt werden, und einen Treiber für Drucker mit Amiga-Zeichensatz gibt es im Verzeichnis „devs/printers“ noch nicht. Man muß daher mit einer PAR:-Datei den Direktkanal öffnen und die erforderlichen Steuercodes im ESC/P-Code übertragen. Die ausführliche Beschreibung dieser Codes befindet sich im Druckerhandbuch zum MPS1200P.

Spezielle Amiga-Zeichen? MPS1200P kennt sie!

Für die Hardcopy eines Listings, welches amigaspezifische Zeichen enthält, muß der Drucker vorher in den Amiga-Modus umgeschaltet werden. Erst dann kann so ein Listing mit LIST, „par:“ am Drucker ausgegeben werden.
Abgesehen von den Sonderzeichen des Amiga ist für mathematische Texte, wegen der griechischen Buchstaben, die fallweise Umschaltung auf den IBM-Zeichensatz erforderlich. Das Demo-Programm auf der Leserdiskette zeigt, wie Umschaltungen des MPS1200P programmiert werden. Die Zeichensätze unterscheiden sich voneinander nur in der nicht genormten Hälfte

```
           KEYBOARD - AMIGA 500
=========================================
             1.Tastenreihe:
` 1 2 3 4 5 6 7 8 9 0 ß ' \    ungeshiftet
~ ! "   § $ % & / ( ) = ? ` |  Shift
` ¹ @ ³ ° ¼ ½ ¾ · ` ' - = \    Alt
~ ! ² # ¢ % ^ & * ( ) _ + |    Shift+Alt
             2.Tastenreihe:
q w e r t z u i o p ü +        ungeshiftet
Q W E R T Z U I O P Ü +        Shift
å • c r t ¤ u ! ø P [ ]        Alt
Å ° c r T ¥ u ¦ Ø P { ]        Shift+Alt
             3.Tastenreihe:
a s d f g h j k l ö ä #        ungeshiftet
A S D F G H J K L Ö Ä #        Shift
æ ß d ` ' ^ ~ ¨ £ ; ' #        Alt
Æ § Ð ` ' ^ ~ ¨ £ : "  #       Shift+Alt
             4.Tastentreihe:
< y x c v b n m , . -          ungeshiftet
> Y X C V B N M , . -          Shift
< ± x c ª º - , , . /          Alt
> ~ / C ª º - ¿ , . /          Shift+Alt
==========================================
   Gedruckt mit den Einstellungen in
  Preferences (Druckertreiber: Epson)
    und ohne zsätzliche Steuercodes.
     Diese Einstellung liefert den
      deutschen Epson-Zeichensatz.
Druckbefehle entweder über PRT: oder LPRINT.
```

```
           KEYBOARD - AMIGA 500
==========================================
             1.Tastenreihe:
` 1 2 3 4 5 6 7 8 9 0 ß ' \    ungeshiftet
~ ! "   § $ % & / ( ) = ? ` |  Shift
` ¹ @ ² ° ¼ ½ ¾ ■ « » - = \    Alt
~ ! ² # ¢ % ^ & * ( ) _ + |    Shift+Alt
             2.Tastenreihe:
q w e r t z u i o p ü +        ungeshiftet
Q W E R T Z U I O P Ü +        Shift
å ° © ® þ ¤ µ ¡ ø ¶ [ ]        Alt
Å ° © ® Þ ¥ µ ¦ Ø ¶ { ]        Shift+Alt
             3.Tastenreihe:
a s d f g h j k l ö ä #        ungeshiftet
A S D F G H J K L Ö Ä #        Shift
æ ß ð ` ' ^ ~ ¨ £ ; ' #        Alt
Æ § Ð ` ' ^ ~ ¨ £ : "  #       Shift+Alt
             4.Tastentreihe:
< y x c v b n m , . -          ungeshiftet
> Y X C V B N M , . -          Shift
< ± × ç ª º — , , . /          Alt
> ¬ ÷ Ç ª º ¯ ¿ , . /          Shift+Alt
===========================================
   Gedruckt mit den Einstellungen in
  Preferences (Druckertreiber: Epson)
und Drucker in den Amiga-Modus geschaltet.
      (über PAR: mit Druckerfile)
     Diese Einstellung liefert den
          Amiga-Zeichensatz
```

```
           KEYBOARD - AMIGA 500
=========================================
             1.Tastenreihe:
` 1 2 3 4 5 6 7 8 9 0 █ ┤ \    ungeshiftet
~ ! "   º $ % & / ( ) = ? ` |  Shift
` ╣ @ │ ░ ╝ ╜ ╛ ╖ ¼ ╗ - = \    Alt
~ ! ▓ # ó % ^ & * ( ) _ + |    Shift+Alt
             2.Tastenreihe:
q w e r t z u i o p ⁿ +        ungeshiftet
Q W E R T Z U I O P ▓ +        Shift
σ ░ ⌐ « ■ ñ ╡ í ° ╢ ╔ ]        Alt
┼ ░ ⌐ « ▌ Ñ ╡ ª ╪ ╢ { ]        Shift+Alt
             3.Tastenreihe:
a s d f g h j k l ÷ Σ #        ungeshiftet
A S D F G H J K L ╥ — #        Shift
µ █ ≡ ` ┤ ^ ~ ¿ ú ; ' #        Alt
╞ º ╨ ` ┤ ^ ~ ¿ ú : "  #       Shift+Alt
             4.Tastentreihe:
< y x c v b n m , . -          ungeshiftet
> Y X C V B N M , . -          Shift
< ▒ ╫ τ ¬ ║ ¡ ╕ , . /          Alt
> ¼ ≈ ╟ ¬ ║ » ┐ , . /          Shift+Alt
==========================================
   Gedruckt mit den Einstellungen in
  Preferences (Druckertreiber: Epson)
und Drucker in den IBM-Modus geschaltet.
      (über PAR: mit Druckerfile)
     Diese Einstellung liefert den
           IBM-Zeichensatz
```

```
           KEYBOARD - AMIGA 500
==========================================
             1.Tastenreihe:
` 1 2 3 4 5 6 7 8 9 0 _ 4 \    ungeshiftet
~ ! "   ' $ % & / ( ) = ? ` |  Shift
` 9 @ 3 0 < = > 7 + ; - = \    Alt
~ ! 2 # " % ^ & * ( ) _ + |    Shift+Alt
             2.Tastenreihe:
q w e r t z u i o p / +        ungeshiftet
Q W E R T Z U I O P \ +        Shift
e 0 ) . ` $ 5 ! x 6 [ ]        Alt
E 0 ) . ^ % 5 & X 6 { ]        Shift+Alt
             3.Tastenreihe:
a s d f g h j k l v d #        ungeshiftet
A S D F G H J K L V D #        Shift
f _ p ` 4 ^ ~ ( # ; ' #        Alt
F ' P ` 4 ^ ~ ( # : "  #       Shift+Alt
             4.Tastentreihe:
< y x c v b n m , . -          ungeshiftet
> Y X C V B N M , . -          Shift
< 1 W g * : - 8 , . /          Alt
> , w G * : / ? , . /          Shift+Alt
===========================================

   Gedruckt mit den Einstellungen in
  Preferences (Druckertreiber: Epson)
und Drucker in den USA-Modus geschaltet.
```

der ASCII-Codes. Für den Vergleich wird daher vom Demo-Programm nur dieser Bereich ausgedruckt.

Den eingestellten Zeichensatz erreicht man auch über die Tastatur. Ob die Anzeige am Bildschirm mit der Ausgabe am Drucker übereinstimmt, hängt von der Druckereinstellung und/oder dem mit Preferences gewählten Treiber ab. Den Unterschied zeigen die Ausdrucke der Tastaturbelegung in verschiedenen Konfigurationen. Sie sind als Eingabehilfe gedacht, damit nicht mit den CHR$-Codes jongliert werden muß, wenn Zeichen gebraucht werden, die im Amiga-Zeichensatz nicht enthalten sind.

(Ilse Wolf)

Farbiger Düsenjäger

Im ständigen Wettkampf: Farbkleckse auf dem Papier und leuchtende Phosphorpunkte auf dem Bildschirm. Ein neuer Farbdrucker hat den Konkurrenzkampf mit der Kathodenröhre aufgenommen.

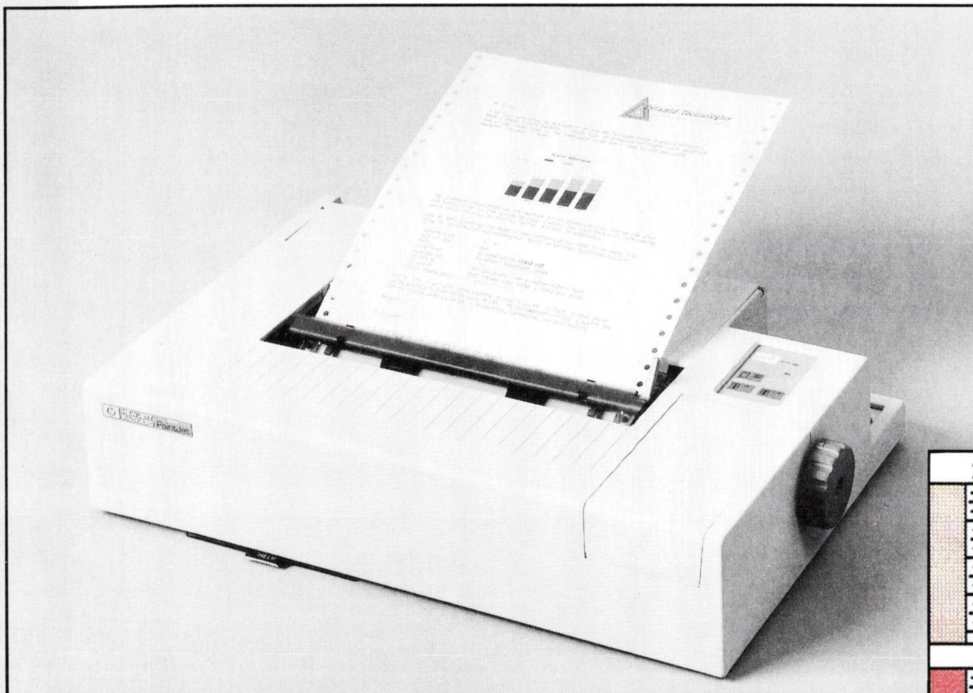
Hohe Anforderungen stellt Hewlett-Pakkard, einer der Marktführer im Bereich hochwertiger Elektronik, an Produkte, die das firmeneigene Logo tragen dürfen. Was die Laserdrucker und Präzisionsplotter berühmt gemacht hat, darf ein kleiner Bruder aus gleicher Werkstatt nicht missen.
Mischen von schwarzer Tinte mit Magenta, Cyan und Gelb ist ein gängiges Verfahren bei Farbdruckern. So auch bei dem Paintjet. In der Farbpalette entsprechen die Blauschattierungen ziemlich genau den Bildschirmfarben, während die Grünvarianten teilweise zu sehr zum Blau tendieren.

Andere Drucker neigen eher dazu, Grün farbecht darzustellen, dafür den Blautönen zu viel Violett beizumischen. Die Tinte wird in zwei Cartridges bereitgestellt: Eine für die schwarze Farbe und eine für die drei anderen Farben. Die Cartridges enthalten ebenfalls die Farbdüsen, die über eine doppelte Reihe von Kontakten aktiviert werden. Diese Einrichtung macht den Paintjet praktisch wartungsfrei — mit jeder neuen Farbpatrone werden neue Düsen eingesetzt. Da sich in der Schwarzpatrone 30 Düsen befinden, während für die drei anderen Farben nur je zehn zur Verfügung stehen, kann ein Farbklecks in schwarz dreimal so groß sein wie in Individualfarben.
Zirka 50 Mark kostet die Schwarzpatrone, die für 1100 Textseiten reichen soll. Die Farbpatrone hat eine Lebensdauer von 180 Seiten mit Grafik und wird für etwa 60 Mark erhältlich sein. Wenn man dann noch die Kosten für das Spezialpapier kalkuliert (zirka 30 Mark für 250 Blatt), kommt man auf 50 bis 70 Pfennig pro bedrucktes Blatt — im Vergleich zu anderen Farbdrucken ein mittlerer Wert. Allerdings muß man mit einrechnen, daß mit jedem Patronenwechsel auch die Farbdüsen erneuert werden.

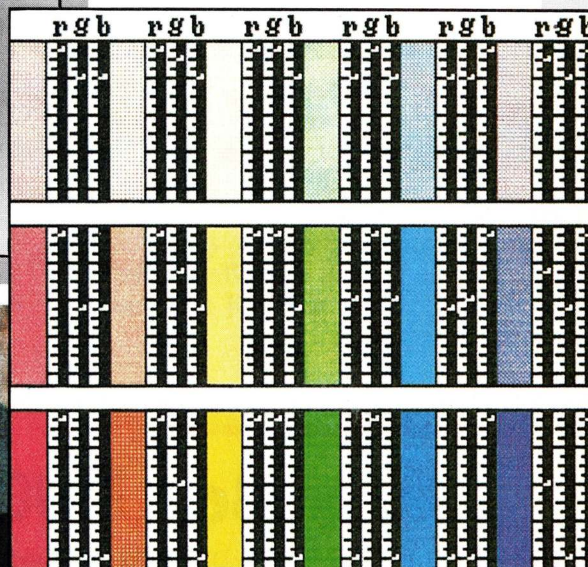
Das Auswechseln der Patronen ist denkbar einfach. An der linken hinteren Ecke verbirgt sich unter einer Klappe die Pumpe. Hier wird die Patrone eingesteckt, und man wartet, bis sich die drei durchsichtigen Behälter mit Farbe gefüllt haben. Danach wird die Patrone in der vorderen Gehäusekammer über den Wischer geführt und in dem vorgesehenen Platz eingerastet. Jetzt sollte ein Selbsttest ausgeführt werden. Treten in dieser Phase der Installation irgendwelche Probleme auf, so liegt das zumeist an verunreinigten Kontakten der Patrone. Mit der Bürste in der vorderen Kammer lassen sich diese leicht reinigen. Im (allerdings englischen) Handbuch ist alles hervorragend dargestellt und illustriert, in Kurzform auch noch mal auf einer Plastikkarte, die aus dem Gehäuse gezogen werden kann.

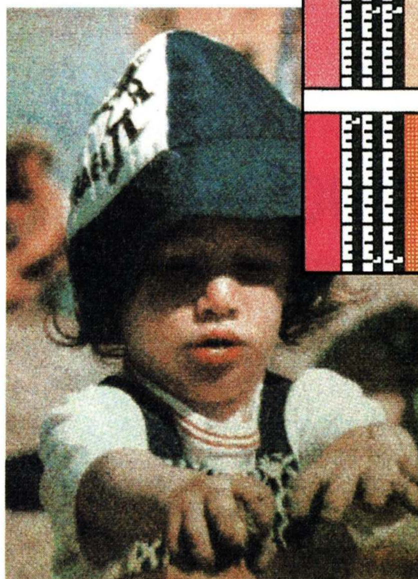
Paintjet akzeptiert Endlospapier ebenso wie Einzelblätter. Sowohl Traktor wie auch Walzenantrieb sorgen für den Papiervorschub. Da der Traktor nicht wegzuschalten ist, lassen sich Einzelblätter nicht beliebig auf der Walze plazieren. Dafür funktioniert der automatische Papiereinzug präzise und zuverlässig. Beste Druckqualität läßt sich mit speziellem Papier erreichen, normales Papier tut jedoch ebenso seine Dienste.

In einer externen Box befindet sich die Stromversorgung. Auf der Kontrolltafel des Druckers sind Netz-, Formfeed-, Top-of-the-Form- und Linefeed-Schalter angebracht. Wenn beim Einschalten der Linefeed-Knopf gedrückt gehalten wird, startet der Drucker den Selbsttest. Mit dem Linefeed-Knopf erhält man auf diese Art einen zweiseitigen Demonstrationsausdruck, während der Top-of-the-Form-Schalter beim Einschalten den Transparentmode für Overheadfolien aktiviert.



Hewlett-Packards farbiger Jetsetter bringt Brillantes aufs Papier





Zwei Beispielausdrucke: ein digitalisiertes Bild und die AmigaWelt-Farbtest-Palette

Je nach Wunsch wird der Drucker mit Centronics- oder RS-232-Schnittstelle geliefert. Nach dem Auspakken, Netz- und Druckeranschluß ist er sofort betriebsbereit. Für erste Druckversuche liegen im Karton 250 Blatt Spezial-Endlospapier, fünf Overheadfolien und ein kompetter Satz Farbpatronen bei. In der höchsten Auflösung von 180 dpi (Dots per Inch) und einer Bildpunktgröße von 0,026 Millimeter lassen sich sehr detailreiche Ausdrucke erzeugen. In dieser Auflösung kann der Paintjet in den vier Farben der Tintenpatronen, wie auch in den Grundfarben Rot, Grün und Blau drucken. In der Auflösung von 90 × 90 dpi sind bis zu 330 Farben realisierbar. Da ▶

der Amiga-Druckertreiber die niedrigere Auflösung nicht beherrscht, werden die 4096 Bildschirmfarben in einer 4 × 4 Matrix im 180 dpi-Modus erzeugt.
Im Gegensatz zu den meisten Nadeldruckern besitzt der Paintjet keinen Draft-Modus; muß er auch nicht, denn seine Near-Letter-Quality-Zeichen bringt er in einer Sekunde 167mal zu Papier (bei 10 cpi). Hewlett-Packard selbst gibt die Druckgeschwindigkeit mit 16,7 Inches pro Sekunde an, was 200 cps bei 12 Zeichen pro Inch (cpi) und 300 Zeichen bei 18 cpi gleichkommt. Während die Textausdrucke bidirektional erfolgen, arbeiten die Düsen bei Grafiken nur in einer Richtung. Als Fonts sind Courier (10 cpi) und für die kleineren Schriften Letter Gothic vorhanden.
Underline- und Boldoptionen werden angeboten, jedoch keine Italics. Dafür ist die Auswahl an nationalen Schriftsätzen reichhaltig. Englisch, Deutsch, Dänisch, Französisch, Italienisch, Norwegisch, Schwedisch und Spanisch sind vorgesehen. Andere Zeichensätze lassen sich mit der Download-Funktion in den 8K-RAM-Buffer einlesen. Subscript und Superscript lassen sich über Wechseln der Schriftgrößen simulieren, da man die vertikale Position pixelweise einstellen kann.

Ausländischer Zeichensatz kein Problem

Leuchtend und feinkörnig sind die Druckergebnisse. Gedrucktes Schwarz könnte allerdings satter sein. Im Transparent-Modus wird dies zwar erreicht, dafür sind die anderen Farben leicht übersättigt. Wenn die neuen Workbench-Versionen 1.3 und 1.4 verbreitet sind, wird sich ein Druckertreiber für den Paintjet auf den Disketten befinden. Mit der 1.2-Variante dauerte ein kompletter Grafikausdruck im Lores-Modus 24 Minuten. Mit der 1.3-Workbench wurden weniger als 5 Minuten gestoppt. Ferner konnte mit dem 1.3-Treiber keine Farbverfälschung bei den Grüntönen festgestellt werden, dafür eine Einschränkung der Farbabstufungen. Für relativ hohe Anfangskosten erhält der Käufer exzellente Text- und Grafikausdrucke. Das hohe Preisniveau aufgrund der Wartungsfreiheit erscheint gerechtfertigt. Ferner beweist ein Blick auf die Preisentwicklung anderer HP-Produkte, daß die Preise nach Markteinführung schnell fallen können.

(Morton A. Kevelson/Übersetzung uk)

Info: Hewlett-Packard, Berner Str. 117, 6000 Frankfurt 56, Telefon: 0 61 72/ 40 00, Preise: 3584 Mark für den Drukker, Schwarzpatrone 58 Mark, Farbpatrone 73 Mark, Spezialpapier 250 Blatt zwischen 28 und 35 Mark, 50 Blatt Transparentfolie zirka 120 Mark

Fortsetzung von Seite 41

Kopierprogramme im Ring

kit“ an. Features wie Normal- und Index-Kopie sind schon selbstverständlich, hinzugekommen ist weiterhin die Parameter-Option. Auf Diskette sind Dutzende von Parametern abgespeichert, weitere werden mit Sicherheit nicht auf sich warten lassen.

„Project D“ beinhaltet aber nicht nur ein „normales“ Kopierprogramm, weiterhin sind noch „Omni Tool“, „Memory Tool“ sowie ein Editor auf Diskette vorhanden. Das Memory Tool gibt jederzeit über den noch zur Verfügung stehenden RAM-Speicher Auskunft, mit dem Omni Tool sind selbst Kopien von Atari-, MS-DOS- und Xenix-Disketten kein Problem.

Das Editor-Tool ist ein kompletter Diskettenmonitor. Anfangs steht die Entscheidung zwischen DOS-Tracks oder MFM-Tracks. In beiden Modi kann bis Cylinder 81 gelesen und geschrieben werden. Im normalen DOS-Format wird der zu editierende Sektor, die Buffersize und die Read-/Write-Length eingegeben. Anschließend kann problemlos der Sektor verändert werden. Als Hilfe ist es möglich, das zugehörige ASCII-Fenster zu öffnen. Die Cursorposition wird in der jeweiligen Byte-, Word- und Longwordnummer stets angegeben. Somit bereitet eine exakte Sektoreditierung keine Probleme. Die MFM-Tracks können genauso einfach untersucht und verändert werden. Nachdem der Sektor eingelesen wurde, können die Größen Word-Sync, Buffersize, Read- und Writelength editiert und auf die Diskette zurückgeschrieben werden. Leider fehlt diesem Tool ein Trackscanner, denn nur damit können die richtigen Werte für eine optimale MFM-Editierung entnommen werden.

TURBO COPY V 2.1

Ein sehr einfach gehaltenes Kopierprogramm, das mit nur einem Modus aufwartet. Nur Normalcopy ist möglich, entweder mit oder ohne Verify. Turbo Copy ist gut geeignet für nicht geschützte Disketten, hat aber bei professioneller Software große Schwierigkeiten. Auch ist nur ein Ziellaufwerk möglich.

WHITE LIGHTNING

Ein Novum im Reigen der Kopierprogramme ist „White Lightning“ von der Firma Versalia. Es ist speziell für Amigas mit einem Laufwerk ausgelegt, es kopiert die Daten zuerst in das RAM und dann erst auf die Zieldiskette. Dadurch ist es auch möglich, Daten mehrfach zu kopieren. Wenn die Option „Multiple Copy“ eingeschaltet ist, fragt der Amiga nach jedem Schreibvorgang, ob die Daten noch einmal geschrieben werden sollen, ansonsten wird ein neuer Block Daten eingelesen. Weiterhin ist es mit „White Lightning“ sogar mit nur 512 KB Hauptspeicher möglich, eine Diskette in einem Arbeitsgang zu kopieren, aber nur, wenn die Quelldiskette nicht allzu viele Files enthält. Dies beruht auf einer speziellen Funktion, die leere Tracks oder Blocks, die viele „Nullen“ beinhalten, auf wenige Bytes komprimiert und so wertvollen Speicherplatz spart.

Nachdem nun die einzelnen Kandidaten ihren Steckbrief abgegeben haben, muß jeder Anwender für sich selbst entscheiden, welches Programm für seine Belange das brauchbarste ist.

Ein Kopierprogramm zu verlangen, das von jedem Original eine Sicherheitskopie anfertigen kann, ist leider utopisch. Solange es Software-Firmen gibt, die auf einen Kopierschutz nicht verzichten wollen, werden die Backup-Tools immer hinterher hinken. Raubkopien — nein danke. Nach Meinung des Autors ist der beste Kopierschutz immer noch ein gutes Handbuch und ein guter Update-Service. Passwortabfrage nach Handbuch ist ebenfalls ein akzeptabler Weg.

(Salus)

Amiga-Copy 1.2
Preis: Data Becker, Düsseldorf, „Das große Floppy Buch“, 59 Mark, im Fachhandel erhältlich

Amiga Tools
Preis: 49,95 Mark, Vertrieb: CSJ, Computersoft GmbH, An der Tiefenriede 27, 5000 Hannover

Fast Lightning
Preis: 59 Mark, Vertrieb: Versalia Versand, Magdalenenweg 4, 4230 Wesel

Marauder II Brainfile 10
Preis: 59 Mark, Vertrieb: Amigaland A. Koppisch, Hohenwaldstraße 26, 6374 Steinbach

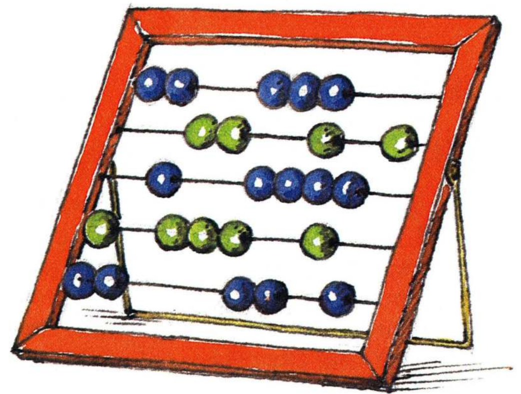
Parameter Copier
Preis: 18 Mark, Vertrieb: Creative Vision, Thomas Lopatic, St.Rochus-Str. 22, 8044 Unterschleißheim

Project D
Preis: 77 Mark, Vertrieb: Soyka Datentechnik, Hattinger Str. 685

Turbo Copy 2.1
Preis: 69 Mark, Vertrieb: Softwareversand Herbert Müller, Dorfstraße 1, 8852 Rain-Unterpeiching

White Lightning
Preis: 59 Mark, Vertrieb: Softwareversand Herbert Müller

Einmaleins der Maschinensprache, Teil 2



In AmigaWelt 2/88 wurden ausführlich die verschiedenen Zahlensysteme erklärt. Diesmal werden die verschiedenen Adressierungsarten des MC 68000 unter die Lupe genommen.

Der Mikroprozessor MC 68000 besitzt verschiedene Adressierungsarten, die sich in fünf Gruppen einteilen lassen: Register indirekt, absolute Adressierung, unmittelbare Adressierung, Register direkt und Adressierung relativ zum Programmzähler.

Register indirekt

Wichtig und häufigste Adressierungsart ist die „indirekte". In der Literatur wird auch der Ausdruck „indexed Adressierung" verwendet. Die Indexregister wurden eingeführt, damit zum Beispiel bei Datensätzen mit verschiedenen Adressen gearbeitet werden kann. Da aber der MC 68000 sehr leistungsstark ist, wurden die speziellen Indexregister abgeschafft und man spricht nur noch von der „indirekten Adressierung". Ein Musterbeispiel ist der Befehl:

MOVE D1,(AR)

Der Inhalt vom Datenregister D1 wird in den Speicherbereich ab der Adresse geladen, die in A4 steht. Das Merkmal der indirekten Adressierung sind die Klammern, da hier nur indirekt auf das Ziel hingewiesen wird. Die Register selbst sind nicht gemeint. Der 68000 verfügt über fünf verschiedene Typen der „indirekten Adressierung":

Adreßregister indirekt

Die „indirekte Adressierung" wird nur bei Adreßregistern angewendet und nicht bei den Datenregistern. Beispiel:

```
MOVE   (A3),D5
```

Der Inhalt der Adresse, auf die A3 zeigt, wird in das Register D5 geschoben oder anders ausgedrückt: Übertrage das Wort, das an der mit A3 adressierten Speicheradresse steht, nach D5. Hat A1 den Wert FFE9BC7A und besitzt diese Speicherzelle den Wert 30, dann wird dieser Wert 30 in das Datenregister D5 geschrieben.

Adreßregister indirekt mit Adreßdistanzwert

Diese Adressierungsart ist vollkommen identisch mit der vorherigen, nur daß hier zum Adreßregister ein beliebiger Wert hinzuaddiert wird. Dieser Wert ist dann der „Adreßdistanzwert" oder auch „Displacement" genannt. Das eigentliche Adreßregister bleibt unverändert. Der Adreßdistanzwert ist eine vorzeichenbehaftete 16-Bit-Zahl im Bereich von −32768 bis +32767. Mit dem Befehl

```
MOVE.L  -450 (A3),D5
```

wird der Inhalt von A3 minus 450 in das Datenregister D5 geschoben. A3 behält seinen ursprünglichen Wert bei.

Adreßregister indirekt mit Adreßdistanzwert und Index

Neu bei dieser Adressierungsart ist der Index, was einem zusätzlichen Adreßregister entspricht. Der Adreßdistanzwert ist diesmal eine vorzeichenbehaftete 8-Bit-Zahl, die im Zweierkomplement dargestellt wird und deshalb im Bereich zwischen −128 und +127 liegen muß. Beispiel:

```
CLR.B  $10 (A4,A5.L)
```

mit folgenden vorgegebenen Werten:

```
Grundregister   A4        00002000
Index           A5        00001990
Displacement              00000010
--------------------------------
Adresse:                  00004000
```

Ab Adresse $00004000 wird nun das Byte gelöscht. Wie man deutlich sieht, ist der Index mit einem Suffix (.L) versehen. Dieser Suffix gibt an, wieviel Bits vom Indexregister zum Grundregister hinzuaddiert werden sollen. Der Adreßdistanzwert kann auch den Wert 0 haben, so daß es nicht notwendig ist, ein Displacement anzugeben.

Adreßregister indirekt mit Postinkrement

der Ausdruck „Postinkrement" läßt sich sehr einfach übersetzen. „Post" bedeutet „nach" und „Inkrement" bedeutet „Erhöhung". Diese Adressierungsart dient also zum automatischen Erhöhen des Adreßregisters nach dem Zugriff. Die Frage ist jetzt nur noch, um wieviel Bytes das Adreßregister nachher erhöht werden muß. Dieses Problem lösen wir gleich anhand eines Beispieles:

```
MOVE.L  (A2)+,D2
```

A2 soll den Inhalt $3000 haben. Der MOVE-Befehl bewirkt nun, daß das Longword, welches sich ab Adresse $3000 im Speicher befindet, in das Datenregister D2 geschoben wird. Anschließend wird der Inhalt von A2 um vier Bytes erhöht, da es sich um eine Longwordadressierung handelt. A2 hat also dann den Inhalt $3004. Bei einer Word-Adressierung (.W) würde das Adreßregister um zwei Bytes und bei einer Byte-Adressierung (.B) um ein Byte erhöht werden. Auf diese Adressierungsart greift man zurück, wenn ein Speicherbereich mit bestimmten Werten aufgefüllt werden soll.

Adreßregister indirekt mit Predekrement
Diese Adressierungsart funktioniert wie die vorherige, der Inhalt des verwendeten Adreßregister wird diesmal vor der Operation um ein, zwei oder vier Bytes (je nach Verwendung der Byte-, Word- oder Longwordadressierung) vermindert.

```
MOVE.L  -(A2),D2
```

A2 soll wieder den Inahlt $3000 haben. Vor Ausführung des Befehls wird nun der Inhalt des Adreßregisters A2 um vier (da Longwordadressierung) vermindert. Das Register A2 enthält also nun die Speicheradresse $2FFC. Der Inhalt dieser neuen Adresse wird dann in das Datenregister D2 geschoben.
Der große Vorteil der beiden Adressierungsarten mit Postinkrement und Predekrement liegt bei der Verwaltung des Stapels oder auch Stack genannt. Was ist eigentlich ein Stack? Ein Stack ist ein Speicher, bei dem Informationen abgelegt und wieder ausgelesen werden können. Dabei muß jedoch beachtet werden, daß ein Stapel nach der LIFO-Methode (Last In, First Out) arbeitet. Dies bedeutet, daß die zuletzt gespeicherte Information auch als erste den Stapel wieder verläßt. Deshalb ist äußerste Vorsicht geboten, bei der Verwendung des Stacks. Auf unserem Schreibtisch legen wir viele Zettel ab, einen auf den anderen. Sobald wir nun anfangen, den Stapel abzubauen, nehmen wir zuerst den obersten Zettel (der als letztes eingegangen ist) und verarbeiten so einen Zettel nach dem anderen. Als letzten Zettel werden wir den untersten bearbeiten, obwohl er aber als erster auf dem Schreibtisch gelandet ist. Genau nach der gleichen Methode (LIFO) arbeitet auch der Stack. Gegenstück dazu ist die Warteschlange (Queue), ein Speicher, bei dem die zuerst abgelegte Information als erste wieder ausgelesen wird. Man spricht von der FIFO-Methode (First In, First Out). Zum Ablegen der Register auf dem Stapel werden folgende Befehle benutzt:

```
MOVE    D0,-(A7) oder
MOVEM   D0-D5,-(A7)
```

Der Befehl MOVEM wird verwendet, um mehrere Register schnell in aufeinanderfolgende Speicherzellen zu legen. Zum Einlesen der Registerinhalte vom Stapel werden folgende Befehle benutzt:

```
MOVE    (A7)+,D0 oder
MOVEM   (A7)+,D0-D5
```

Absolute Adressierung

Absolut lange Adressierung
Der Operand selbst ist eine 32-Bit-Adresse. Der Prozessor MC 68000 rechnet zwar mit 32-Bit-großen Adressen, dafür adressiert er aber mit nur 24 Bits. Somit kann maximal ein Adreßbereich von 2^24 = 16777216 Bytes, entspricht 16 MBytes (1 MByte = 1024 Bytes), direkt angesprochen werden.

```
MOVE    $0244FE04,D1
```

Der Inhalt der Speicherzelle $0244FE04 wird in das Datenregister D2 geladen. Grundregel: Greift der Prozessor bei Word- oder Longwordoperationen auf den Speicher zurück, so dürfen nur gerade Adressen mit den Endziffern 0, 2, 4, 6, 8, A, C, E benutzt werden! Mit dem Befehl „EVEN“ kann jederzeit ein Füllbyte (0) eingefügt werden. Ebenso lassen sich symbolische Adressen verwenden. Beispiel: Man bezeichnet die Speicherzelle $0244FE04 als DEMO und gibt nur noch den Befehl MOVE DEMO,D2 an. Der Assembler fügt dann selbst die zugehörige Adresse ein.

Absolut kurze Adressierung
Der Assembler unterscheidet selbständig zwischen „kurzer“ oder „langer“ Variante. Bei „Absolut kurze Adressierung“ ist der Operand eine 16-Bit-Adresse. Damit kann man entweder auf die ersten 32 Kbytes oder auf die höchsten 32 KBytes im Speicher zugreifen. Wissenswert ist hier lediglich, daß bei den Adressierungsregistern der unbenutzte Teil immer mit dem Inhalt des ersten Bits des benutzten Teils aufgefüllt wird. Man spricht deshalb auch von der Vorzeichenerweiterung, da das erste Bit das Vorzeichen enthält. Ein typischer Befehl für eine „absolut kurze Adressierung“:
NOT.L $4000
Mit diesem Befehl wird das Einerkomplement des Longwords gebildet, das sich ab Adresse $4000 im Speicher befindet.

Unmittelbare Adressierung
Diese Adressierungsart, auch „immediate“ genannt, benutzt den Operanden hinter dem Befehl direkt als Wert selbst.
MOVE #50,D1
Der Prozessor schiebt den Wert 50 unmittelbar in das Datenregister D1. Zu beachten ist hier, daß diese „unmittelbaren“ Operanden je nach Art des Befehls und des Registers zwischen 3 und 32 Bit (3, 8, 16, 32) lang sein können. Als Erkennungszeichen gilt das vorangestellte Doppelkreuz (#). Eine weitere wichtige Assemblerregel sollte jeder Programmierer beachten: Bei jedem Assemblerbefehl mit zwei Operanden gilt, daß zuerst die Quelle und dann das Ziel angegeben wird. Bei einer eventuellen Verknüpfung wird das Ergebnis immer im Ziel abgespeichert.

```
MOVE  #30,D0
Add   #20,D0
```

Zuerst wird das Register D0 mit dem Wert 30 geladen. Anschließend wird der Wert 20 zum Register D0 hinzuaddiert. Im Register D0 steht jetzt nicht mehr der Wert 30 oder 20 sondern der Wert 50. Desweiteren ist es möglich auf die „schnelle unmittelbare Adressierung“ zurückzugreifen. Der „unmittelbare“ Operand darf hier nicht länger als 8 Bits sein:

```
MOVEQ  #30,D0
```

Dies bedeutet MOVE QUICK. Schiebe schnell den Wert 30 in das Datenregister D0.

Register direkt
Bei der direkten Registeradressierung können Datenregister, Adreßregister und das Statusregister verwendet werden. Der Operand in einem angegebenen Register ist als effektive Adresse enthalten.

Datenregister
Als Register können acht 32-Bit-Datenregister (D0 bis D7) verwendet werden. Der Befehl NOT D2 bildet das Einerkomplement für das Wort im Register D2. Falls zwei effektive Adressen benuzt werden, ist es möglich, verschiedene Adressierungsarten zu kombinieren: ▶

```
MOVE  D2,$1244
```

Hier liegt eine Kombination der direkten Datenregisteradressierung und der absolut kurzen Adressierung vor. Mit diesem Befehl werden die 16 unteren Bits von D2 in die Adresse $1244 geschoben. Natürlich kann das Datenregister auch als Ziel dienen:

```
MOVE  $1244,D2
```

Das Wort, das sich ab Adresse $1244 im Speicher befindet, wird in das Register D2 geschoben.

Adreßregister

Als Register können sieben verschiedene Adreßregister (A0 bis A6) verwendet werden.

```
MOVE  $1244,A3
```

Das Adreßregister A3 ist hier das Ziel einer Übertragung. Das Adreßregister kann natürlich auch als Quelloperand benutzt werden:

```
MOVE.L  A1,D1
```

Das Longword, welches sich im Register A1 befindet, wird in das Register D1 geschoben. Auch hier handelt es sich wieder um eine Kombination zweier Adressierungsarten, Adreßregister direkt und Datenregister direkt.

Statusregister

Folgende Befehle finden hier Verwendung in Verbindung mit dem Statusregister:

```
- ANDI    (logische AND-Verknüpfung
          unmittelbarer  Daten mit
          einem zweiten Operanden)
- EORI    (logische EOR-Verknüpfung ....)
- ORI     (logische OR-Verknüpfung ....)
```

Folgender Befehl veranschaulicht die Struktur:

```
ANDI   #$0020,SR
```

Die logischen Operationen haben wir bereits im ersten Teil unseres Kurses besprochen, so daß hier nicht näher darauf eingegangen wird.

Relative Programmzähleradressierung

Programcounter-relative (PCR) Adressierung mit Adreßdistanzwert: Diese Adressierungsart ist identisch mit der Adressierungsart „Adreßregister indirekt mit Adreßdistanzwert"). Als Adreßregister wird hier nur der Programmzähler (PC) benutzt. Der Adreßdistanzwert (Displacement) ist wieder eine vorzeichenbehaftete 16-Bit-Zahl, also eine Zahl zwischen −32768 und +32767:

```
CLR    $2000 (PC)
```

Wäre der Inhalt des Programmzählers gleich $1000, so löscht der Prozessor den Inhalt der Speicherstelle $3000.

PCR-Adressierung mit Adreßdistanzwert und Index

Auch hier genügt ein Hinweis auf die indirekte Adressierung. Der Adreßdistanzwert ist wieder eine vorzeichenbehaftete 8-Bit-Zahl, also eine Zahl zwischen −128 und +127.

Somit haben wir das Kernstück der Assemblerprogrammierung kennengelernt. Als Abschluß dieses Kurses soll ein Übersichtsplan helfen, die verschiedenen Adressierungsarten genauer kennenzulernen. Im nächsten Teil unseres Kurses werden wir uns dann genauer mit den Verzweigungsbefehlen beschäftigen.

(Stephan Quinkertz)

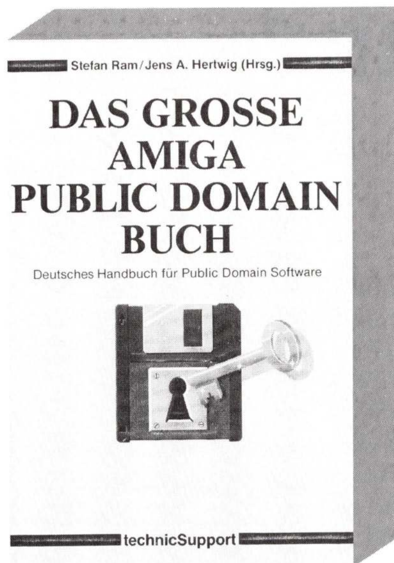
Adressierungsart	Schreibweise	Erklärung
Datenregister direkt	Dn	EA = Dn
Adreßregister direkt	An	EA = An
Statusregister direkt	Sr	EA = Sr
Adreßregister indirekt (ARI)	(An)	EA = (An)
ARI mit Displacement	d16 (An)	EA = (An) + d
ARI mit Displacement und Index	d8 (An,Rn)	EA = (An) + d + (Rn)
ARI mit Postinkrement	(An)+	EA = (An) und An = An + n , n = 1,2,4
ARI mit Predekrement	-(An)	EA = (An) und An = An - n , n = 1,2,4
Absolut kurz	$XXXX	EA = 16-Bit-Adresse
Absolut lang	$XXXXXXXX	EA = 24-Bit-Adressse
Unmittelbar	# Daten	Operand ist Datenwert
PCR mit Displacement	d16 (PC)	EA = (PC) + d16
PCR mit Displacement und Index	d8 (PC,Rn)	EA = (PC) + d + (Rn)

mit:

EA	effektive Adresse
Dn	beliebiges Datenregister
An	beliebiges Adreßregister
Rn	beliebiges Adreß- oder Datenregister
d8	8-Bit-Displacement
d16	16-Bit-Displacement

Adressierungsarten im 68000-Assembler

DER SCHLÜSSEL ZUR SCHATZTRUHE:
DAS GROSSE AMIGA PUBLIC DOMAIN HANDBUCH
Band I und Band II



S. Ram/J.A. Hertwig (Hrsg.)
DAS GROSSE AMIGA PUBLIC DOMAIN BUCH,
ISBN 3-926847-01-8, DM 49,-

Deutsches Handbuch zur preiswerten Public Domain Software
Das GROSSE AMIGA PUBLIC DOMAIN HANDBUCH ist der Schlüssel zur Schatztruhe von fast 900 PD-Disketten. In der Public Domain findet jeder Anwender das richtige Programm: ob nun Spiele und Grafiksoftware, ob Utilities, Formatierer, Kopierprogramme, Demos oder Programmierhilfen. Public Domain bietet einfach alles!

Das deutsche Handbuch öffnet die PD-Schatztruhe für viele Anwender. In Band I dieser Reihe sind 44 wichtige PD-Programme mit genauen Anleitungen beschrieben. Z.B. proff, life, AmigaTeX, hack, kermit, units, vt100, amcat, arc, setfont, record, replay, make, PipeHandler, PrintText, MenuBuilder, uShow, DPSlide, PrtDrvGen, blitz, gfxmem, disksalv und viele andere Programme.

Darünber hinaus bringt Band I: Einführung in die PD-Software, CLI-Hilfen, Erste Schritte mit PD, Beschreibung zum Editor ED.

Komplette Liste der wichtigsten PD-Reihen
Alle **FISH-Disks** bis 128, die Reihe von **FAUG** und **PANORAMA** sind übersichtlich gelistet - damit Sie Programme leicht finden.

technicSupport Marketing und Verlag, Bundesallee 36-37, 1000 Berlin 31, Telefon 030-8621314/5

NEU ab AUGUST 1988: Band II
DAS ZWEITE
AMIGA PUBLIC DOMAIN
BUCH
Hrsg. R. Leithaus/J.A.Hertwig
ISBN 3-926847-05-0, DM 49,-

Band II der PD-Reihe von technicSupport bringt z.B. deutsche Anleitungen zu folgenden Programmen:
A-Render, FontEdit, Modula II, Gonf, LOGO, Quickcopy, PopCLI, WordWrite, AmigaMonitor, Arp, DirUtil II, VirusX, Browser, Zoo, M-CAD, viele Spiele, Utilities und Grafikprogramme. Dazu: Informationen über Workbench 1.3 sowie 1.4, neue PD-Reihen (z.B. RPD) u.v.m.

SCHRIFTSATZ auf AMIGA mit:
AmigaTeX

Schriftsatz für alle Zwecke: Bücher, Broschüren, wissenschaftliche oder mathematische Texte, Zeitschriften. Ausgabe über alle Druckertypen oder über Satzbelichter.
Beispiel:

$$\int_0^\infty \frac{t-ib}{t^2+b^2} e^{iat}\, dt = e^{ab} E_1(ab),$$

$$\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) \left|\varphi(x+iy)\right|^2 = 0$$

$$\pi(n) = \sum_{m=2}^{n} \left\lfloor \left(\sum_{k=1}^{m-1} \lceil m_k \rceil \right)^{-1} \right\rfloor$$

Originalausdruck mit QMS PS 810

Ausführliche Beschreibung, Musterausdrucke mit verschiedenen Druckern und 2 Demodisketten anfordern.
Info-Script + 2 Demo-Disks DM 20,-

BESTELLUNG Hiermit bestelle ich bei technic-Support:, Bundesallee 36-37, 1000 Berlin 31, T. 030-8621314/5

x DAS GROSSE AMIGA PUBLIC DOMAIN BUCH, Bd. I ISBN 3-926847-01-8	DM 49,-
x DAS ZWEITE AMIGA PUBLIC DOMAIN BUCH ISBN 3-926847-05-0	DM 49,-
x Band I und Band II zum Kombi-Super-Preis	DM 89,-
x 10 Disketten mit 44 Programmen zu Band I	DM 114,-
x AmigaTeX Info-Script + 2 Demo-Disks	DM 20,-
x 2 PD-Katalog-Disketten	DM 20,-

Alle Bestellungen + DM 5,- Versandkostenanteil.
Ich bezahle per .... Nachnahme, per .... Verrechnungsscheck

Name: ....

Straße: ....

Ort: ....
Datum: .... **Unterschrift**

Anzeige mit AMIGA 2000 gesetzt

Grafik-Kurs in Bits und Bytes — Teil 2

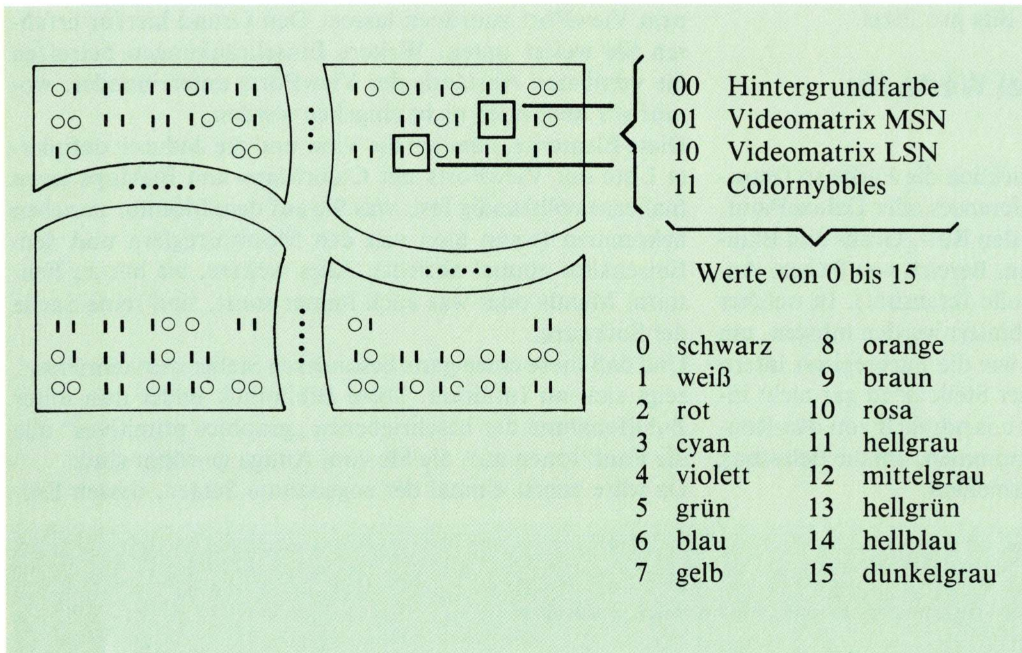
Über Bibliotheken und Intuition

In Heft 2/88 wurde anhand zweier Beispielprogramme ein unbeschwerter Einstieg in die Grafikprogrammierung vermittelt (Beispiel 2, ein einfaches Zeichenprogramm, steht Abtippern erst mit diesem Heft zur Verfügung). Diesmal erfolgt ein Überblick über die internen Strukturen und die Zusammenhänge zwischen Elementen der Bibliotheken, Grafik, Layers und Intuition.

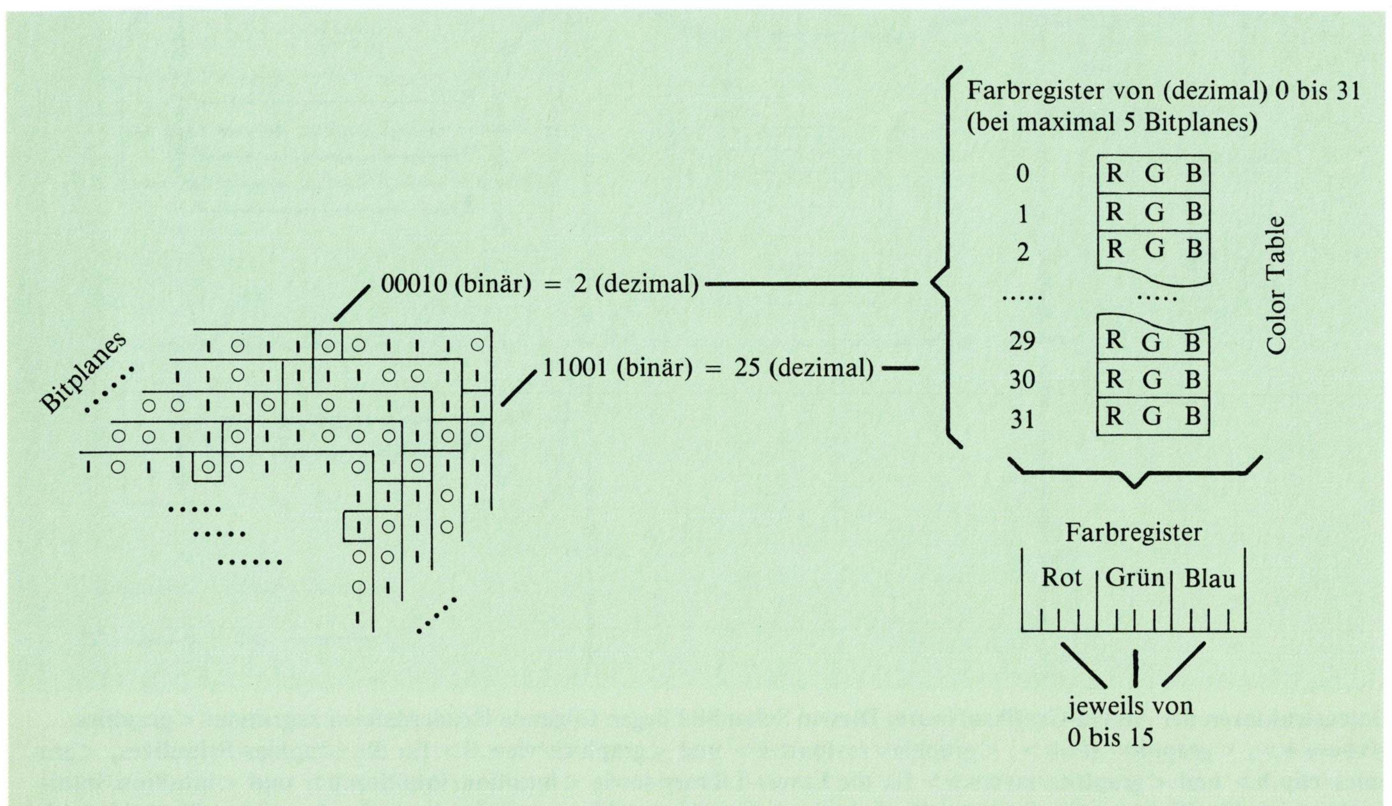
Als bekannt wird vorausgesetzt, daß Sie wissen, was ein Screen (Bildschirm) und ein Window (Fenster) ist. Beide sind fundamentale Elemente der Benutzer-Schnittstelle (User-Interface) des Amiga, „Intuition" genannt, und sollten daher wohl auch dem unbedarften Einsteiger ein Begriff sein.
Doch vor der Erläuterung dieser luxuriösen Features beschäftigen wir uns zuerst mit den sogenannten „graphics primitives", den Grafikfunktionen der untersten Ebene. Versuchen Sie, die im Text beschriebenen Elemente des Amiga-Betriebssystems anhand der Abbildungen zu verfolgen und sich so die Beziehungen der Bestandteile der Amiga-Grafiksoftware untereinander klarzumachen.
Zentrales Glied der „graphics.library", die man als direkt der Hardware (den Grafikchips also) übergeordnete Institution ansehen kann, ist die zugehörige Library-Base mit dem Namen „GfxBase", die Ihnen noch aus dem letzten Heft in Erinnerung sein sollte. Die Bedeutung einer Library-Base können Sie sich in AmigaWelt 1/88 im Artikel „Back to the Roots: Assemblerprogrammierung auf dem Amiga" vergegenwärtigen.
Diese Struktur ist genau einmal vorhanden, und hierin befindet sich ein Zeiger auf die sogenannte „View", was man am treffendsten mit „Ansicht" übersetzt. Diese View legt das fest, was Sie auf dem gesamten Bildschirm — und damit ist diesmal die Oberfläche Ihres Monitors oder Fernsehers gemeint — sehen. Zwar können mehrere View-Strukturen innerhalb des Amiga existieren, jedoch ist nur eine davon zu einem bestimmten Zeitpunkt aktiv (die ActiView) und bestimmt die Darstellung auf dem Monitor. Dies ist üblicherweise die von Intuition festgelegte View, aber ... darauf kommen wir noch zurück.
Nun kennen die Grafikchips des Amiga ja recht verschiedene Modi der Darstellung: niedrige Auflösung (LORES von „LOw RESolution"), hohe Auflösung (HIRES, HIgh RESolution), der Hold-And-Modify-Modus (HAM), in dem sich alle 4096 Farben zur gleichen Zeit darstellen lassen, und viele weitere mehr. Auch die Anzahl und Auswahl an Farben differiert und muß entsprechend festgelegt werden. Dies geschieht nicht in der „View", sondern in den sogenannten „ViewPorts".
Ein „ViewPort" ist ein Bereich, in dem ein bestimmter Satz an Darstellungsmodi, Farben etc. konstant festgelegt ist. Wie Ihnen durch die Screens bekannt sein wird, beherrscht der Amiga die gleichzeitige Darstellung völlig verschiedener Modi. Farbregister, Art der Darstellung und BitMap sind also lokal für jeden ViewPort. Doch was ist eine BitMap? Recht einfach: Diese legt die Speicherbereiche (Planes = Ebenen) fest, in denen die eigentliche Grafik abgelegt ist, ähnlich etwa einem der 8 KB großen Blöcke im HIRES-Modus des C64. Während beim C64 jedoch uneingeschränkt (bei konstantem Wert aller Bytes der Videomatrix) nur zwei verschiedene Farben gleichzeitig dargestellt werden konnten, nämlich entsprechend dem Zustand des zugehörigen Bits in der Bitmap, so werden beim Amiga mehrere solcher Ebenen miteinander kombiniert, um die gewohnte Farbenvielfalt zu erreichen. Reicht Ihre C64-Erinnerung noch weiter zurück, so ist Ihnen vielleicht auch noch der Multicolor-BitMap-Modus ein Begriff: Hier wurden jeweils ZWEI BITS miteinander kombiniert und so eine uneingeschränkte Darstellung von vier Farben möglich. Mit „uneingeschränkt" ist hier gemeint, daß wirklich jedem einzelnen Pixel eine bestimmte Farbe zugeordnet werden kann. Durch geschickte Aufteilung der Grafik und entsprechende Belegung von Videomatrix und Colornybbles wurden natürlich wesentlich farbenfrohere Darstellungen erreicht.

Bitplane — so lautet die neue Adresse für Grafiken

Wie Sie noch sehen werden, macht die Kombination von Betriebssystem-Software und Custom-Chip-Funktionen, speziell Blitter und Copper, den Begriff der BitMap beim Amiga freilich ungleich umfassender als beim C64.
Beim Amiga wurde anstelle dieses „Pixel Addressing" des C64 vielmehr das „Bitplane Addressing" verwendet. In jeder der bereits erwähnten Bitplanes entspricht ein Bit einem Pixel auf dem Bildschirm. Um nun verschiedene Farben darstellen zu können, werden alle zu einem Pixel gehörigen



Pixel-Addressing im Multicolor-Bitmap-Modus des C64



Bitplane-Addressing beim Amiga

Bits quer durch die Planes miteinander kombiniert, um daraus zu bestimmen, welche Farbe dem Pixel zukommt. Im Falle von drei Planes ist dies dann eine von acht Farben, da die jeweils drei Bits eines Bildschirmpunktes (Pixels) acht verschiedene Zustände annehmen können. Das daraus gebildete Binärwort bestimmt eines der Farbregister im Bereich von (dezimal) 0 bis 7. Der Vorteil des „Bitplane Addressing" gegenüber des „Pixel Addressing" besteht in der effizienteren Ausnutzung des Speichers bei solch „krummen" Zahlen wie 3 oder 5 Bits pro Pixel.

Rot, Grün, Blau — drei Werte für Farbenvielfalt

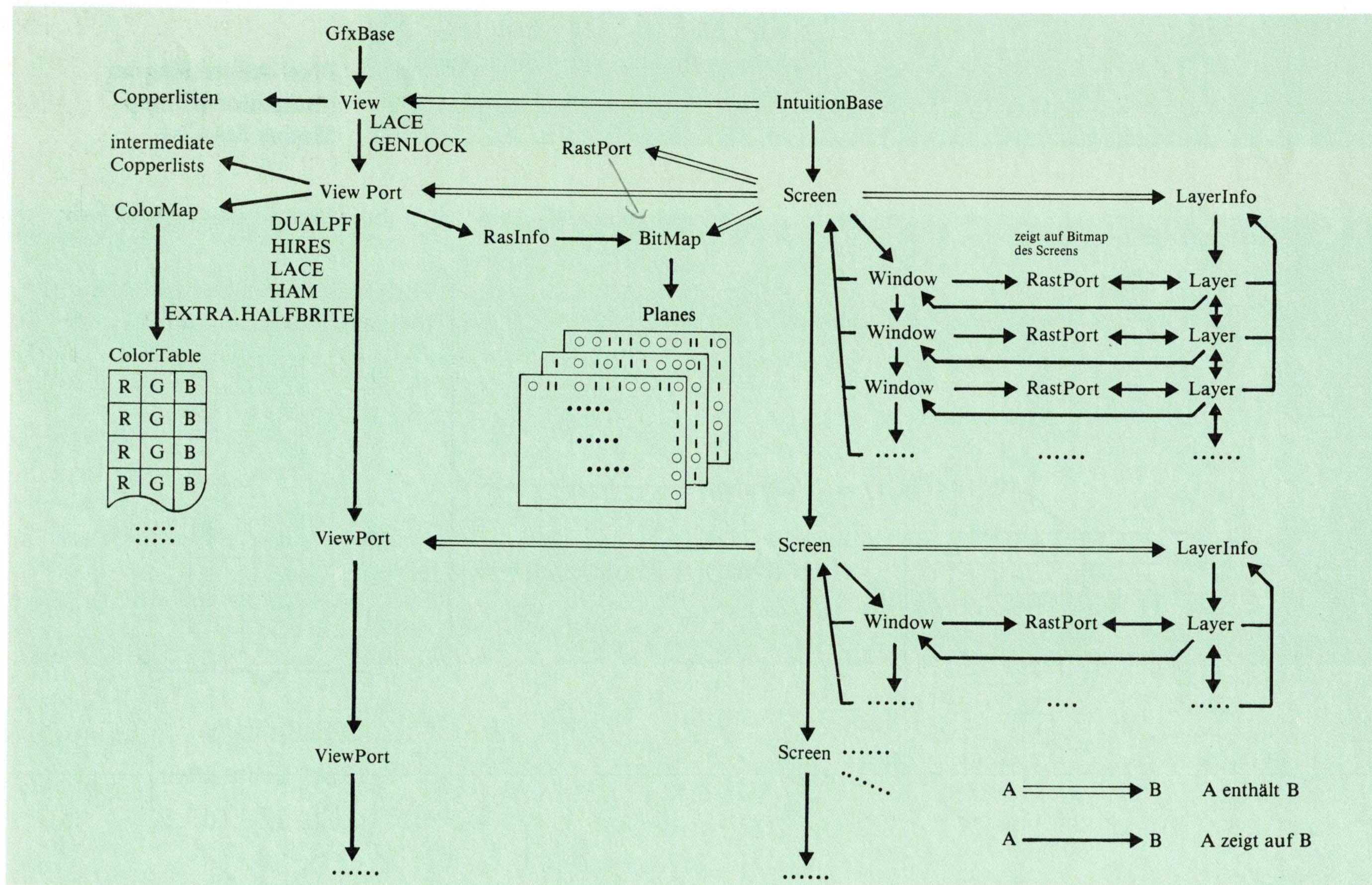
In den Farbregistern wird nun wirklich die Farbe so festgelegt, wie wir es, etwa aus den Preferences oder DeluxePaint, gewohnt sind, nämlich zerlegt in den Rot-, Grün- und Blauanteil. Jeder Anteil bewegt sich im Bereich von 0 (kein Anteil dieser Grundfarbe) bis 15 (volle Intensität). In welcher Reihenfolge die Planes nun kombiniert werden müssen, um den Farbwert zu errechnen oder wie die Farbregister intern angeordnet sind, soll uns an dieser Stelle noch gar nicht interessieren: All diese Arbeit wird uns nämlich von den Routinen der Grafik-Bibliothek abgenommen, um die Belastung mit internen Darstellungen zu vermeiden.

Doch zurück zu den ViewPorts: Diese bilden nun eine einfach verkettete Liste in ihrer Reihenfolge auf dem Bildschirm (= innerhalb der View) von oben nach unten. Hieran läßt sich eine Einschränkung erkennen, denen ViewPorts in der gegenwärtigen Version des Amiga aufgrund der Hardware noch unterliegen: Sie können NUR UNTEREINANDER angeordnet werden. So ist es also nicht möglich, zwei ViewPorts nebeneinander oder gar überlappend zu definieren. Jede horizontale Rasterzeile muß sich höchstens einem ViewPort zuordnen lassen. Den Grund hierfür erfahren Sie weiter unten. Weitere Einschränkungen betreffen die vertikalen Abstände der ViewPorts untereinander, worauf wir aber noch nicht eingehen werden.

Diese Elemente, nämlich die View und die dadurch definierte Liste der ViewPorts mit ColorMaps und BitMaps legen (nahezu) vollständig fest, was Sie auf dem Monitor zu sehen bekommen (wenn man von den Monitorreglern und dem Einschalter einmal absieht). Alles weitere, bis hin zu Fenstern, Menüs oder was auch immer sonst, sind reine Sache der Software.

Und daß diese einen ganz besonderen Stellenwert einnimmt, zeigt sich an Intuition. Diese Bibliothek bildet nun unter Zuhilfenahme der beschriebenen „graphics primitives" alle die Funktionen aus, die Sie vom Amiga gewöhnt sind:

Da wäre zuerst einmal der sogenannte Screen, dessen Ent-



Datenstrukturen der Amiga-Grafiksoftware: Diesem Schaubild liegen folgende Headerdateien zugrunde: <graphics/gfxbase.h>, <graphics/gfx.h>, <graphics/rastport.h> und <graphics/view.h> für die Graphics-Primitives, <graphics/clip.h> und <graphics/layers.h> für die Layers-Library sowie <intuition/intuition.h> und <intuition/intuitionbase.h> für die Anwender-Schnittstelle „Intuition". Graphics und Layers sind weitestgehend autark, während Intuition auf beide Libraries zugreift.

sprechung zum ViewPort der „graphics primitives“ Sie sicherlich gleich zu Beginn erkannt haben. Anhand des Schaubildes können Sie sehen, wie jedem Screen ein ViewPort zugeordnet ist; der „struct ViewPort“ ist sogar ein Teil des „struct Screen“, ebenso wie andere Strukturen. Ein „struct Screen“ besitzt also nicht einen Zeiger auf den ViewPort, sondern enthält ihn gleich vollständig. Doch wollen wir die Verkettungen vom vergleichbaren Ansatz verfolgen, wie bei der „graphics.library“ auch, von der Basis:
Die IntuitionBase enthält die View, der Pointer „ActiView“ in GfxBase zeigt also mitten in die IntuitionBase hinein. IntuitionBase zeigt auf den ersten einer ebenfalls einfach verketteten Liste von Screens, die in der Reihenfolge von „vorne“ nach „hinten“, wenn man jeden Screen als eine der „Schichten“ in der Darstellung auf dem Monitor, die sich gegenseitig verdecken, sehen will, angeordnet sind.
Die Screen-Struktur enthält nun auch die BitMap (nicht jedoch die Planes) und den RastPort, die untereinander diverse Zeiger aufeinander unterhalten. Die Bedeutung des RastPorts sollte Ihnen noch aus dem letzten Heft bekannt sein: Dies ist der Parameter, der bei den Zeichenfunktionen der Grafik-Bibliothek festlegt, auf welche Zeichenfläche die Funktion wirken soll. Nicht nur der Fläche eines Windows, wie in den beiden ersten Beispielprogrammen, entspricht ein solcher RastPort, sondern auch dem gesamten Screen. Wie, glauben Sie, werden wohl die ScreenGadgets gezeichnet?

Grafikspezialist entlastet CPU

Die Rolle des COPPERS

Der Copper ist, ähnlich der 68000, ein Mikroprozessor, der bestimmte Befehle ausführen kann, jedoch sind seine Funktionen speziell auf die Programmierung der Grafikfunktionen abgestimmt. Sie werden in seinem Befehlssatz vergeblich nach komplexen Adressierungsarten oder etwa arithmetischen Funktionen suchen.
Sicherlich haben Sie sich bei der Erklärung der ViewPorts gefragt, wie es denn erreicht wird, daß mitten im Bildschirm plötzlich auf eine ganz andere Art der Darstellung umgeschaltet werden kann, andere Speicherbereiche (Planes) nun als Quelle der Grafikdaten dienen und weiteres mehr. Sollten Sie einmal den C64 programmiert haben, so werden Sie sich vielleicht daran erinnern, wie mit dem Zeilenstrahl-Interrupt einige trickreiche Effekte programmiert werden konnten: Da wurde ein mehrfarbiger Rahmen erzeugt, mehr als acht Sprites dargestellt und diese sogar im Rahmen sichtbar gemacht, und auf diese Weise überzog Text und Grafik die gesamte Monitorfläche. All dies wurde dadurch erreicht, daß im geeigneten Augenblick die Werte in den Registern des VIC geändert wurden.
Nicht anders arbeitet nun der Copper, jedoch ohne den Hauptprozessor zu belasten. Der Copper führt Befehle in Abhängigkeit von der Position des Zeilenstrahls aus und lädt so die Farbregister, die maximal sechs Zeiger auf die Planes, die Bits für die Display-Modes und weitere Register mehr von neuem. Dies geschieht während der meist drei Rasterzeilen in den Hintergrundfarben der benachbarten ViewPorts, die zwei Screens voneinander trennen. Hieraus ▶

läßt sich ersehen, daß der Copper durchaus einige in ihrer Gesamtheit zeitaufwendige Befehle ausführen muß, bis der neue ViewPort seinen Daten entsprechend in den Registern repräsentiert ist, weswegen Screens auch nur vertikal verschoben werden können.
Diese Befehlsanreihung, also das Programm, das der Copper bei jedem Rasterstrahldurchlauf neu abarbeitet (für die Profis: So spart man Latches!), nennt man auch „Copperliste", jedoch wird dieser Begriff unglücklicherweise ohne nähere Spezifizierung auch für „User-Copperlisten" und andere „intermediate Copperlists" verwendet. Allgemein läßt sich jedoch sagen, daß eine Copperliste die Befehle für den Copper definiert. Wird nun ein Screen gezogen oder in der „Tiefe" neu angeordnet, so stockt meistens kurz die gesamte Aktivität auf dem Amiga oder verlangsamt sich zumindest etwas, kein Wunder, müssen doch mehrere dieser Copperlisten völlig neu berechnet werden, eine sehr rechenintensive Angelegenheit.
Nun kann auch noch eine nicht ganz korrekte Angabe zur Erklärung der Darstellungsmodi der ViewPorts korrigiert werden: Es wurde gesagt, daß diese nicht in der View, sondern im ViewPort festgelegt werden, was im Normalfall auch stimmt. Eine wichtige Ausnahme bildet jedoch der sogenannte INTERLACE-Modus, der sich Ihnen mit seinem entnervenden Geflimmere sicherlich schon vorgestellt hat. Wenn Sie einen Screen solcher Darstellung nach unten gezogen haben, dann wird Ihnen aufgefallen sein, daß auch dahinterliegende Screens plötzlich dem gleichen Effekt unterliegen, auch wenn diese nur eine vertikale Ausdehnung von beispielsweise 200 Zeilen normaler Auflösung nutzen. Allgemeiner ausgedrückt läßt sich sagen, daß genau dann, wenn mindestens ein sichtbarer Screen im Interlace dargestellt wird, dies für das gesamte Display, oder — wie Sie mittlerweile ahnen werden — die View gilt. Display-Modi mit globalen Auswirkungen auf die gesamte View müssen also auch in dem entsprechenden Feld der View festgehalten werden. Der Grund hierfür liegt in der Art und Weise, wie ein Interlace-Bild intern repräsentiert und vom Monitor dargestellt wird. Für Details muß ich Sie bis zu dem Zeitpunkt vertrösten, da Copperlisten unser Thema sind.
Damit sind auch schon die grundlegenden Verbindungen zwischen Intuition und Graphics geklärt, weswegen wir nun zur Verbindung zwischen Intuition und Layers kommen.

Layers — Zerlegung einer BitMap in virtuelle Schichten

Vorherrschendes Element auf einem Screen ist das sogenannte Window, ein, wie anhand der Workbench sichtbar, scheinbar vom Screen unabhängiger Bestandteil der Darstellung, der sich verschieben, in der Größe ändern oder auch in der „Tiefe" zu anderen Fenstern anordnen läßt.
Nun ist Ihnen sicherlich anhand der Erklärungen zum Copper klargeworden, daß nicht einfach zum Zeitpunkt, da der Rasterstrahl die linke obere Ecke eines Fensters erreicht hat, auf eine neue BitMap, die hypothetische BitMap des Windows, umgeschaltet werden kann: Dies wäre einfach zu zeitaufwendig. Zu jedem ViewPort/Screen existiert demnach genau eine BitMap, die kontinuierlich abgebildet wird. Wenn Sie sich die durch eine solche BitMap definierten Planes genau ansehen könnten, so würden Sie feststellen, daß der Inhalt derselben genau dem entspricht, was Sie in diesem Screen auf dem Schirm sehen können, unabhängig von Windows, Menüs etc. (Ich habe es auch noch nicht explizit kontrolliert, aber lassen Sie uns einfach einmal blind daran glauben ...).
Es muß also, unterschiedlich zu den ViewPorts, ein anderer Weg gefunden werden, eine BitMap in scheinbare „Schichten" aufzuteilen. Dabei kommt man nicht darum herum, wirklich Speicherbereiche der Planes gegeneinander auszutauschen, zu füllen und weiteres mehr, während bei den Screens ja nur der Zeitpunkt der Darstellung variiert wurde. Hierfür ist die sogenannte „layers.library" (layer = Schicht) zuständig. Sie „multiplext" eine BitMap, teilt sie also in verschiedene Zeichenflächen auf, die, wie im Falle der Workbench, sogar von verschiedenen Programmen (genauer: Tasks) zur gleichen Zeit benutzt werden können. Ebenfalls ist sie für das sogenannte „Clipping" zuständig, dafür, daß eine Schreib-Operation nur innerhalb eines definierten Bereichs stattfindet, also etwa nicht über die Grenzen eines Fensters hinausgeschrieben wird. Auch muß in bestimmten Fällen, etwa wenn ein anderes Objekt einen Teil des Ausgabefensters verdeckt, die Ausgabe in den nicht sichtbaren Bereich, in bestimmte „Backup-Regionen" (Regions) umgelenkt werden; dazu später noch mehr.

Fast menschliche Allüren: Bitmaps benötigen ständig Erfrischungen

Ebenso wie die Screens, so bilden auch die Windows eine verkettete Liste in der Reihenfolge ihrer Tiefenanordnung; der Screen zeigt also auf das am weitesten „vorne" befindliche Fenster, jenes also, das auch nach Verschiebung nicht verdeckt werden kann.
Ein Fenster (mit Ausnahme des selten verwendeten Typs „GimmeZeroZero") ist also genau einem Layer zuzuordnen. Während ein Layer mehr die interne Verwaltung beschreibt, konzentriert sich die Window-Struktur auf Dinge wie die Titelleiste, Rahmen, Menü und ähnliches.
Wird nun ein Fenster derart positioniert, daß es ein anderes überlappt, so wird, bevor der Speicherbereich des verschobenen Fensters in seinen neuen Bereich kopiert wird, der verdeckte Ausschnitt in einen separaten Bereich gerettet, um im Falle der Wiedersichtbarmachung zurückkopiert zu werden. Dies kostet natürlich extra Speicherplatz, erspart dem Programmierer jedoch viele Unannehmlichkeiten. Dieser Modus wird daher als „SMART REFRESH" bezeichnet, weil smarterweise automatisch etwas für die Rettung des Fensterinhalts getan wird, ohne daß der Programmierer in irgendeiner Weise damit belästigt wird. Im Unterschied hierzu wird bei „SIMPLE REFRESH" der verdeckte Bereich NICHT gerettet. Prominente Vertreter des letzteren Modus sind TxEd oder AmigaBASIC, wobei sich bei letzterem zeigt, wie entnervend es für den Benutzer sein kann, wenn ein ohnehin langsames Programm auch noch selbständig den Refresh durchführen muß. Zurück zum SMART REFRESH: Anschließend werden solche Bereiche wieder aus den „Backup-Regionen" zurückgeholt, die nach der

Verschiebeoperation (oder nach dem Schließen eines Fensters) nicht mehr verdeckt werden. Werden von einem Task Schreiboperationen in solche verdeckten Regionen ausgeführt, so erfolgt ein automatischer Update, so daß diese immer auf dem aktuellen Stand sind.

Daß diese Zerlegung eines Fensters in viele kleine Teilbereiche aufwendig und ab einer bestimmten Anzahl an Fenstern auch durchaus stark verlangsamend wirkt, ist sicherlich vielen Anwendern bekannt: Nicht selten ruckeln beim Scrolling in einem Fenster mehrere Dutzend kleine Ausschnitte. Weiterhin stoppen bestimmte Operationen, etwa die Anzeige eines Menüs oder das manuelle Verschieben oder Vergrößern eines Fensters, jegliche Ausgabe in anderen Layers. Hierfür ist die sogenannte „Layer_Info" verantwortlich. Sie ist allen Layers einer BitMap übergeordnet, ähnlich dem Verhältnis zwischen View und ViewPorts. Möchte ein Task eine spezielle Operation mit einem Layer durchführen, so muß vorher die Layer_Info „verriegelt" werden, um Konflikte auszuschließen. So wird der Rahmen, der die neue Position beim Verschieben eines Fensters bezeichnet, durch Invertierung innerhalb der Screen-BitMap erzeugt und durch weitere Invertierung wieder gelöscht. Dies würde nicht funktionieren, wenn der Rahmen zwischenzeitlich übermalt werden könnte.

Kommen wir noch zu einem wichtigen Unterschied zwischen dem RastPort eines Windows und eines Screens. Wie anhand der Grafik zu erkennen, ist letzterem kein Layer zugeordnet. Da die Layers-Library nicht nur für die Aufteilung einer BitMap zuständig ist, sondern, wie erwähnt, auch für das Clipping, muß bei der direkten Ausgabe über den RastPort eines Screens mit Vorsicht gearbeitet werden, da hier fehlgeleitete Zeichenoperationen leicht den Rechner zum Absturz bringen können. Dafür ist eine solche Ausgabe allerdings auch schneller, denn natürlich kostet die Überwachung durch die Layers-Library Zeit. Darauf werden wir im nächsten Heft noch weiter eingehen, ebenso wie auf Farben.

Der Blitter ist das Arbeitspferd der Grafiksoftware: Nicht so flexibel wie der Hauptprozessor, der manche Vorbereitung leisten muß, bis der Blitter endlich loslegen kann, dafür aber unglaublich schnell, wenn er erst einmal gestartet wurde. Seine Hauptanwendung liegt im Kopieren und Verknüpfen von Speicherbereichen, im Linienziehen und im Flächenfüllen, wobei vom schnellen Kopieren speziell die Layers-Library profitiert, doch auch die Dekodierung der im MFM-Format aufgezeichneten Diskettensignale erledigt er prompt. Mit bis zu drei Quellbereichen können binäre Operationen durchgeführt werden, die dann über einen vierten DMA-Kanal gespeichert werden.

Dazu kommen noch Verschiebungen über den sogenannten „Barrel-Shifter", eine Gatter-Funktion, deren Verschiebegeschwindigkeit (im Gegensatz zur 68000 etwa) nicht von der Anzahl der Verschiebepositionen abhängt, und automatische Randbegrenzungen.

Was sich mit diesen auf den ersten Blick des Nicht-Eingeweihten recht unscheinbar ausnehmenden Funktionen alles anfangen läßt, kann man auf Fish-Disk 31 sehen: Hier wird unter Zuhilfenahme des Blitters das Spiel „Life" in bisher nicht erreichter Geschwindigkeit implementiert. Riskieren Sie ruhig einmal einen Blick hinein, auch in den Quelltext — es ist reines C!

Und fragen Sie doch einmal einen Atarianer, warum der wesentlich primitivere Atari-Blitter so lange bis zum fertigen Produkt brauchte, wenn Jay Miner den Amiga-Blitter gar schon heute als veraltet bezeichnet.

Nach diesem mehr theoretischen Abschnitt sollten Sie die grundlegenden Elemente der grafischen Darstellung auf dem Amiga einander zuordnen können und sie in Beziehung zu Hardware und Software setzen können.

Sprites, wie auch der Mauszeiger eines darstellt, wurden in den obigen Ausführungen nicht berücksichtigt, ebenso wie andere wichtige Elemente der Grafikprogrammierung, so etwa durch ein Anwenderprogramm definierte Copperlisten, Graphic Elements (GELs) und Animationen sowie vieles weitere mehr. Das Modell wird im Laufe des Kurses aber noch weiter verfeinert werden. *(Ralph Babel)*


Auf Leserdiskette



Überprüfen mit checksum

Einfaches Zeichenprogramm

Sprache: C

```
/*                                                          653
** AmigaWelt-Grafikkurs: Beispiel 2                         653
** Copyright (C) 11-Feb-1988 by Ralph Babel                 653
** all rights reserved - alle Rechte vorbehalten            653
*/                                                          358
                                                            358
/*** included files ***/                                    580
                                                            580
#include <exec/types.h>                                     916
#include <exec/ports.h>                                     399
#include <libraries/dos.h>                                  244
#include <graphics/gfxbase.h>                               023
#include <graphics/gfxmacros.h>                             096
#include <graphics/rastport.h>                              258
#include <intuition/intuition.h>                            706
#include <intuition/intuitionbase.h>                        093
                                                            093
/*** types ***/                                             275
                                                            275
typedef int bool;                                           930
                                                            930
/*** external symbol references ***/                        576
                                                            576
#ifndef __ARGS                                              050
#ifdef AZTEC_C                                              756
#define __ARGS(a) ()                                        088
#else                                                       284
#define __ARGS(a) a                                         844
#endif                                                      157
#endif                                                      574
                                                            574
struct Library *OpenLibrary __ARGS((char *, ULONG));        322
VOID CloseLibrary __ARGS((struct Library *));               343
struct Window *OpenWindow __ARGS((struct NewWindow *));     645
VOID CloseWindow __ARGS((struct Window *));                 048
VOID SetAPen __ARGS((struct RastPort *, ULONG));            073
VOID SetDrMd __ARGS((struct RastPort *, ULONG));            281
VOID Move __ARGS((struct RastPort *, LONG, LONG));          508
VOID Draw __ARGS((struct RastPort *, LONG, LONG));          170
VOID WritePixel __ARGS((struct RastPort *, LONG, LONG));    867
struct Message *WaitPort __ARGS((struct MsgPort *));        532
struct Message *GetMsg __ARGS((struct MsgPort *));          418
VOID ReplyMsg __ARGS((struct Message *));                   695
VOID printf __ARGS((char *, ));                             320
                                                            320
STATIC VOID Line __ARGS((struct RastPort *, WORD, WORD,
 WORD, WORD, UBYTE,                                         936
 UBYTE));                                                   110
STATIC bool MouseInWindowP __ARGS((struct Window *,
 struct IntuiMessage *));                                   546
                                                            546
/*** global symbols ***/                                    192
                                                            192
struct GfxBase *GfxBase;                                    874
struct IntuitionBase *IntuitionBase;                        728
                                                            728
/*** constants ***/                                         470
                                                            470
#define COLOR 1                                             548
                                                            548
```

```
/*** data section ***/                                         970
                                                               970
STATIC struct NewWindow NW =                                   639
 {                                                             238
 160, 64,          /* SHORT LeftEdge, TopEdge; */              333
 320, 68,          /* SHORT Width, Height; */                  930
 -1, -1,           /* UBYTE DetailPen, BlockPen; */            084
 SIZEVERIFY                                                    272
 | MOUSEBUTTONS                                                036
 | MOUSEMOVE                                                   996
 | CLOSEWINDOW,    /* ULONG IDCMPFlags; */                     700
 WINDOWSIZING                                                  828
 | WINDOWDRAG                                                  682
 | WINDOWDEPTH                                                 927
 | WINDOWCLOSE                                                 649
 | SMART_REFRESH                                               318
 | REPORTMOUSE                                                 023
 | RMBTRAP                                                     639
 | NOCAREREFRESH,  /* ULONG Flags; */                          540
 NULL,             /* struct Gadget *FirstGadget; */           553
 NULL,             /* struct Image *CheckMark; */              326
 "Beispiel 2",     /* UBYTE *Title; */                         951
 NULL,             /* struct Screen *Screen; */                988
 NULL,             /* struct BitMap *BitMap; */                489
 80, 32,           /* SHORT MinWidth, MinHeight; */            792
 ~0, ~0,           /* USHORT MaxWidth, MaxHeight; */           314
 WBENCHSCREEN      /* USHORT Type; */                          919
 };                                                            347
                                                               347
/*** code section ***/                                         753
                                                               753
STATIC bool MouseInWindowP(w, im)                              311
struct Window *w;                                              564
struct IntuiMessage *im;                                       442
 {                                                             773
 return im->MouseX >= w->BorderLeft && im->MouseX <
  w->Width - w->BorderRight                                    292
  && im->MouseY >= w->BorderTop && im->MouseY < w->Height
   - w->BorderBottom;                                          660
 }                                                             513
                                                               513
STATIC VOID Line(rp, startx, starty, endx, endy, mode,
 color)                                                        130
struct RastPort *rp;                                           797
WORD startx, starty, endx, endy;                               992
UBYTE mode;                                                    161
UBYTE color;                                                   036
 {                                                             591
 SetDrPt(rp, 0xffff);                                          223
 SetDrMd(rp, mode);                                            029
 SetAPen(rp, color);                                           626
 Move(rp, startx, starty);                                     240
 Draw(rp, endx, endy);                                         707
 }                                                             124
                                                               124
/*** entry point ***/                                          930
                                                               930
LONG main()                                                    064
 {                                                             955
 struct IntuiMessage *im, IM;                                  255
 struct RastPort *rp;                                          066
 struct Window *w;                                             411
 BOOL running;                                                 939
 LONG result;                                                  589
 WORD startx, starty, endx, endy;                              784
 enum modes { IGNORE, PIXEL, LINE } mode;                      815
 BOOL oldline;                                                 377
                                                               377
 result = RETURN_FAIL;                                         579
                                                               579
 if((GfxBase = (struct GfxBase *)                              707
  OpenLibrary("graphics.library", LIBRARY_VERSION)) !=
  NULL)                                                        908
  {                                                            687
  if((IntuitionBase = (struct IntuitionBase *)                 331
   OpenLibrary("intuition.library", LIBRARY_VERSION)) !=
   NULL)                                                       434
   {                                                           669
   result = RETURN_ERROR;                                      077
                                                               077
   if((w = OpenWindow(&NW)) != NULL)                           661
    {                                                          524
    rp = w->RPort;                                             871
    mode = IGNORE;                                             376
    oldline = FALSE;                                           886
                                                               886
    for(running = TRUE; running; )                             879
     {                                                         310
     if(WaitPort(w->UserPort) != NULL)                         872
      {                                                        219
      while(running                                            533
       && (im = (struct IntuiMessage
        *)GetMsg(w->UserPort)) != NULL)                        127
       {                                                       774
       IM = *im;                                               200
       ReplyMsg((struct Message *)im);                         842
       switch(IM.Class)                                        251
        {                                                      338
        case MOUSEBUTTONS:                                     616
         switch(IM.Code)                                       002
          {                                                    149
          case MENUDOWN:                                       652
           if(mode == IGNORE && MouseInWindowP(w, &IM))        287
            {                                                  806
            mode = LINE;                                       687
            startx = IM.MouseX;                                734
            starty = IM.MouseY;                                747
            oldline = FALSE;                                   401
            }                                                  242
           break;                                              394
          case SELECTDOWN:                                     580
           if(mode == IGNORE)                                  163
            mode = PIXEL;                                      202
           break;                                              186
          case MENUUP:                                         098
           if(mode == LINE)                                    161
            {                                                  168
            if(oldline)                                        451
             {                                                 890
             Line(rp, startx, starty, endx, endy,
              COMPLEMENT, 0xff);                               788
             oldline = FALSE;                                  042
             Line(rp, startx, starty, endx, endy, JAM1,
              COLOR);                                          896
             }                                                 533
            mode = IGNORE;                                     246
            }                                                  275
           break;                                              891
          case SELECTUP:                                       270
           if(mode == PIXEL)                                   287
            mode = IGNORE;                                     624
           break;                                              408
          }                                                    333
         /* drop through! */                                   259
        case MOUSEMOVE:                                        585
         switch(mode)                                          029
          {                                                    500
          case PIXEL:                                          500
           if(MouseInWindowP(w, &IM))                          752
            {                                                  707
            SetDrMd(rp, JAM1);                                 549
            SetAPen(rp, COLOR);                                658
            WritePixel(rp, IM.MouseX, IM.MouseY);              208
            }                                                  781
           break;                                              269
          case LINE:                                           115
           if(oldline)                                         946
            {                                                  133
            Line(rp, startx, starty, endx, endy,
             COMPLEMENT, 0xff);                                683
            oldline = FALSE;                                   337
            }                                                  410
           if(MouseInWindowP(w, &IM))                          942
            {                                                  081
            Line(rp, startx, starty, endx = IM.MouseX,
             endy = IM.MouseY,                                 732
             COMPLEMENT, 0xff);                                828
            oldline = TRUE;                                    455
            }                                                  944
           break;                                              216
          }                                                    885
         break;                                                077
        case CLOSEWINDOW:                                      105
         running = FALSE;                                      477
         break;                                                373
        }                                                      878
       }                                                       299
      }                                                        964
     }                                                         369
    CloseWindow(w);                                            654
    }                                                          435
   else                                                        948
    printf("Unable to open window.\n");                        107
   CloseLibrary((struct Library *)IntuitionBase);              789
   }                                                           142
  else                                                         703
   printf("Cannot open intuition!?!\n");                       721
  CloseLibrary((struct Library *)GfxBase);                     237
  }                                                            158
 else                                                          975
  printf("Cannot open graphics?!?\n");                         005
                                                               005
 return result; /* exit(result); */                            177
 }                                                             378
```

Listing zum C-Zeichenprogramm aus AmigaWelt 2/88

```
;                                                              059
;Beispiel2.bat                                                 606
;                                                              793
FailAt 1                                                       403
LC1 -b1 -ceft -oT: Beispiel2                                   863
LC2 -r1 -s -v T:Beispiel2                                      396
BLink LIB:c.o+T:Beispiel2.o TO Beispiel2 LIB
 LIB:lc.lib+LIB:amiga.lib SC SD ND                             443
Delete T:Beispiel2.o                                           740
FailAt 10                                                      694
```

Batch-File zum C-Listing „Zeichenprogramm“

Leser helfen Lesern

Oftmals sind es Kleinigkeiten, die den Umgang mit dem Rechner erleichtern. Aber erstmal draufkommen! Wer kleine Tricks entdeckt hat, sollte sie daher nicht für sich behalten. Die User werden es danken und wir mit mindestens 50 Mark für einen AmigaKick.

$019 Maus und Joystick an Port Eins

Viele Spielprogramme benötigen einen Joystick in Port Eins. Eine kleine Schaltung macht Umstecken überflüssig: Joystick und Maus werden gleichzeitig angeschlossen und per Schalter aktiviert.
Zum Aufbau der Schaltung werden nur wenige Bauteile benötigt, das Schaltungsprinzip ist einfach: Am Port Eins des Amiga wird ein Kästchen mit zwei Joystickbuchsen angeschlossen. Per Kippschalter wird ein Relais aktiviert, das eine der beiden Buchsen auf Port Eins schaltet. So kann zwischen Maus und Joystick ausgewählt werden. Der Aufbau der Schaltung erfolgt gemäß Abbildung 1 und dürfte auch für wenig geübte Lötkolbenartisten kein Problem sein.
(Oliver Rost, Andreas Wagner)

Bauteileliste:
2 Subminiaturstecker Serie D 9-polig (männlich)
1 Subminiaturstecker Serie D 9-polig (weiblich)
1 5-Volt-Relais mit 5 Umschaltkontakten
1 Ein-/Ausschalter
(Bauteilepreis ohne Gehäuse etwa 15 Mark)

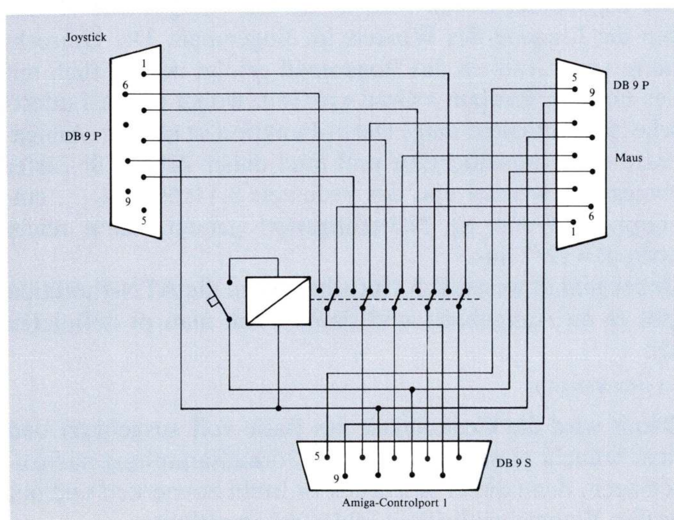


Abbildung 1: Schaltplan der Port-Umschaltung.

$01A Variantenreicher Ausdruck übers CLI

Jetzt läßt sich direkt übers CLI Schriftart und Zeichensatz auf bequeme Art bestimmen. Ob Listingausdruck, Brief oder Rohmanuskript: Ausdruckformate sind mit der folgenden Vorgehensweise blitzschnell festgelegt.
Als Lösung bieten sich kurze Dateien an, in denen die gewünschte Escape-Sequenze steht. Mit TYPE lassen sich diese Dateien an den Drucker übermitteln. Mit einer Textverarbeitung, die Escapes darstellen kann, läßt sich eine derartige Datei erzeugen. Es geht auch mit folgendem kurzen BASIC-Programm. Das Programm eröffnet eine Datei mit dem entsprechenden Namen auf Diskette, schreibt den gewünschten Escape-Code in die Datei und schließt sie. Als Beispiel hier ein Programm, welches zwei Dateien erzeugt. Die Datei NLQ.esc schaltet den Drucker auf NLQ-Modus, die Datei NORMAL.esc schaltet wieder in Normalschrift zurück. Die angegebenen Escape-Sequenzen gelten für den STAR NL10 und müssen für andere Drucker abgeändert werden (nach Handbuch).

```
OPEN "NLQ.esc" FOR OUTPUT AS #1
                  Eröffnen der Datei
PRINT#1,CHR$(27);CHR$(120);CHR$(1)
                  NLQ einschalten
CLOSE             Datei schließen

OPEN "NORMAL.esc" FOR OUTPUT AS #1
                  Eröffnen der Datei
PRINT#1,CHR$(27);CHR$(120);CHR$(0)
                  NLQ ausschalten
CLOSE             Datei schließen
```

Um nun den Escape-Code zum Drucker zu schicken, benutzt man den Befehl

```
TYPE >par: NLQ.esc
```

Wichtig ist hierbei, daß man den Drucker über die Parallel-Device anspricht, da man direkt mit Escape-Sequenzen arbeitet. Man kann natürlich auch mit einem Execute-File arbeiten, um die Sache universeller zu machen. Die Datei hierzu, die zum Beispiel den Namen DRUCK hat, könnte folgendermaßen aussehen:

```
.K NAME
TYPE >par: <NAME>.esc
```

Mit dem Aufruf

```
Execute DRUCK NLQ
```

wird der Drucker auf NLQ-Modus umgeschaltet, und mit

```
Execute DRUCK Normal
```

geht er wieder in den Normalmodus zurück. ▶

Auf gleiche Art läßt sich jede beliebige Druckart festlegen. Natürlich ist es möglich, mehrere Escape-Sequenzen in eine Datei zu schreiben, was folgendes kurze Programm zeigt. Es schaltet den Drucker auf Eliteschrift, setzt den linken Rand auf 10, was Lochen und Einheften ermöglicht, und schaltet den US-Zeichensatz ein:

```
OPEN "TEST.esc" FOR OUTPUT AS #1
                          Eröffnen der Datei
PRINT#1,CHR$(27);CHR$(77)
                          ELITE ein
PRINT#1,CHR$(27):CHR$(108);CHR$(10)
                          Linker Rand auf 10
PRINT#1,CHR$(27);CHR$(52);CHR$(0)
                          US-Zeichensatz
CLOSE                     Datei schließen
```

Nach dem Aufruf mit

```
Execute DRUCK TEST
```

ist der Drucker dann ideal zum Ausdruck von Listings, C-Sourcecodes oder Directories eingestellt.

(Dieter Kuntz)

$01B Screendump in eigenen Programmen

Auf der Diskette „Amiga-Extras 1.2" ist in der Schublade „Basicdemos" das Programm „Screenprint" enthalten. Löscht man den Demo-Vorspann bis zum Beginn der eigentlichen Hardcopy-Routine „Screendump" und speichert diese als ASCII-File, so läßt sich der „Screendump" mit MERGE an eigene Programme anhängen. Der Aufruf kann dann mit „GOSUB Screendump" erfolgen.
Für die einwandfreie Funktion sind einige Dinge zu beachten, die im Kommentar des Originals nicht enthalten sind:

1) In den Preferences der Workbench müssen die für den Grafik-Dump erforderlichen Parameter, wie Treshold, Black and White beziehungsweise Color, Image und so weiter eingestellt sein.

2) Die Paletten 0 und 1 (Hinter- und Vordergrund müssen richtig „gemischt" sein (siehe Demo-Teil von Screenprint). Werden die Standard-Paletten belassen und Hinter- und Vordergrund nur mit COLOR bestimmt, so muß im Programm eine kleine Änderung erfolgen, und zwar in der IF-Sequenz, in der abgefragt wird, ob der Fensterrahmen gedruckt werden soll:

```
IF BorderFlag% = 0 THEN  'Kein Rahmen
srcX% = srcX% + 4
srcY% = srcY% + 4
srcWidth%  = srcWidth%  - 4 - 11
scrHeigth% = scrHeight% - 11 - 4
END IF
```

Gegenüber dem Original wurden die Werte von scrX% und scrY% (das sind „Rastport Beginn X und Y") um 1 erhöht.
Geschieht das nicht, so wird der seitliche Fensterrahmen nicht unterdrückt. Einfacher ist es daher die Paletten 0 und 1 passend zu mischen.

Tritt während der Hardcopy des Rastports ein Fehler auf, so wird die Ausgabe mit einem Fehlercode abgebrochen. Nachfolgend dazu die sieben Fehlercodes im Klartext:

1 Druckvorgang durch Anwender abgebrochen: Beispiel: Bei einem Requester „Printer Trouble" wurde auf das Cancel-Feld gedrückt anstatt den Fehler zu beheben.

2 Drucker nicht grafikfähig: Beispiel: Typenrad-Drucker.

3 Hold-And-Modify kann nicht invertiert werden.

4 Druck-Koordinaten sind nicht erlaubt: Beispiel: Die Anfangskoordinaten des Rastports liegen außerhalb desselben. Kann hier nicht auftreten — außer es wurden unzulässige Programm-Änderungen vorgenommen.

5 Druck-Dimensionen sind nicht erlaubt: Beispiel: Die angegebene Breite und Höhe des Rastports ist größer als der aktuelle Rastport. Auch das kann hier theoretisch nicht passieren.

6 Für interne Berechnungen ist kein Speicher frei

7 Für Druckerpuffer ist kein Speicher frei

$01C pi mal Daumen

Die trigonometrischen Funktionen des AmigaBasic verlangen die Eingabe des Winkels im Bogenmaß. Die Umrechnung vom Grad- in das Bogenmaß erfolgt bekanntlich mit der Formel: Radiant = Grad * pi/180, wobei pi die Ludolfsche Zahl enthalten muß. Die pi-Funktion ist aber im Amigabasic nicht implementiert und muß daher Ziffer für Ziffer eingegeben werden und das bedeutet: 3.141592654. . . einzutippen. Wenn ein Näherungswert genügt, dann reicht auch pi = 22/7 aus.
Dabei geht es wesentlich einfacher, denn die ATN-Funktion gibt es im Amigabasic und damit kann man pi definieren als:

```
pi=4*ATN(1)
```

Damit wird die Genauigkeit des Basic voll ausgenutzt und man braucht pi nicht erst in einer Formelsammlung nachzuschlagen, denn dieser Ausdruck ist leicht zu merken und mit größer Wahrscheinlichkeit fehlerfrei eingetippt.
(I. Wolf)

$01D Prioritäten richtig setzen

Wenn Sie oft rechenintensive Programme im Hintergrund laufen lassen und im Vordergrund mit interaktiven Programmen wie einem Editor weiterarbeiten, dann empfiehlt es sich, die Vordergrundpriorität auf den Wert 1 hinaufzusetzen (siehe Change Taskpri im DOS-Handbuch). Da ein Editor die meiste Zeit damit verbringt, auf eine Eingabe vom Benutzer zu warten, wird dadurch der Hintergrundtask (etwa ein Compiler) nicht verlangsamt. Andererseits reagiert der Editor schneller als sonst auf eine Tasteneingabe, da die Ausführung des Hintergrundtasks sofort auf Eis gelegt wird, wenn für das Vordergrund-CLI ein Ereignis, in diesem Fall der Tastendruck, anliegt. Bei gleicher Priorität wird die Ausführung des Tasks erst unterbrochen, wenn das normale Quantum der Zeitscheibe abgelaufen ist.
(Ralph Babel)

$01E Kopieren

Wenn Dateien vom oder ins aktuelle Arbeitsverzeichnis (Current Directory) kopiert werden sollen, tun sich die meisten ein wenig schwer, da es unter AmigaDOS, im Unterschied zu MS-DOS, kein Symbol für dieses Verzeichnis gibt, während „/" für das Parent, das übergeordnete Verzeichnis, und „:" für das Root gilt.
Das Template von COPY lautet

FROM,TO/A,ALL/S,QUIET/S

Das heißt, daß „FROM" gar nicht angegeben werden muß! Wollen Sie vom aktuellen Verzeichnis in ein anderes kopieren, so können Sie eingeben . . .

COPY TO Ziel

Etwas schwieriger wird es, wenn in das Current Directory kopiert werden soll, was uns zum Problem des „Symbols" zurückbringt. Ich werde es Ihnen ausnahmsweise verraten, ohne Sie vorher mit Syntax und Systematik zu nerven: Es ist der Leerstring!

COPY DF2: TO „" ALL

. . . kopiert also den Inhalt von DF2: ins aktuelle Verzeichnis.

Probleme mit dieser Methode gibt es, wenn eine Datei (und kein Verzeichnis) auf diese Weise kopiert werden soll, da dieses um eine Ebene zu weit „oben" landet. Im Root wird der Kopiervorgang verweigert. Dies muß ein konzeptioneller Fehler im COPY-Befehl sein, in der ARP-Version funktioniert es ohne Beanstandung.
Ein Nebeneffekt des Pattern-Matching läßt sich ausnutzen, wenn Sie eine Datei kopieren und dabei die genaue Schreibweise beibehalten wollen, die Sie natürlich gerade nicht wissen: Setzen Sie den Namen in Klammern. Dadurch wird der Name als Pattern angesehen und der COPY-Befehl verwendet die Originalschreibweise. Dies funktioniert mit allen Patterns, also auch #?.
Anstelle von . . .

COPY EinSehrSeltsamerName TO DF2:

. . . einfach . . .

COPY (einsehrseltsamername) TO DF2:

... oder, noch einfacher, ...

COPY ein# TO DF2:

... letzteres in der Hoffnung, daß nicht noch mehr Dateien mit „ein" beginnen. *(Ralph Babel)*

$01F ECHO — Ausgabe mit Pfiff

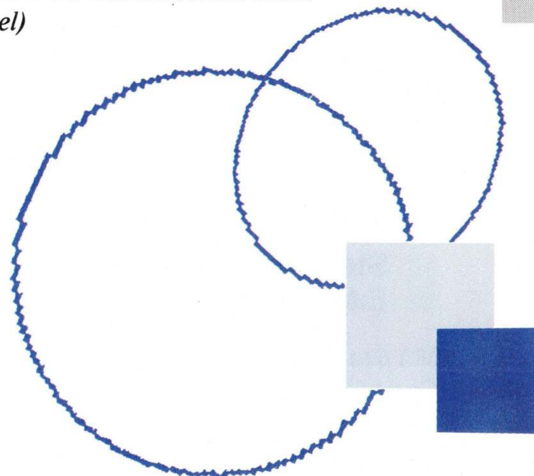
Ein nichtdokumentierter Nebeneffekt der in BCPL geschriebenen CLI-Kommandos besteht darin, daß in Anführungszeichen gesetzte Argumente bestimmte Steuerzeichen enthalten können. Diese sind den üblichen BCPL-Konventionen angelehnt, ein (versteckter) Hinweis hierauf befindet sich auf Seite 242 des AmigaDOS-Manuals (alte Ausgabe zur 1.1-Version, letzter Satz zum DiskEd — The Disk Editor — WAS? Noch einer?).
Escape-Sequenzen werden von '*' eingeleitet:

*" für "
** für *
*N für CHR(10) LineFeed = NewLine
*E für CHR(27) Escape

Großschreibung ist nicht zwingend notwendig, so daß Sie auch „*n" und „*e" schreiben können.
Hierdurch lassen sich nun alle die speziellen ANSI-Steuercodes im ECHO-Befehl anwenden. Geben Sie hierzu einfach einmal ein . . .

ECHO „*E[H*E[J"

Diese Anweisung löscht den Bildschirm! Genauso läßt sich der Cursor ein- und ausschalten sowie Zeichenfarbe und -modus, Randsteller und Cursorposition festlegen. Ziehen Sie hierzu Ihr DOS-Handbuch zu Rate.
(Ralph Babel)



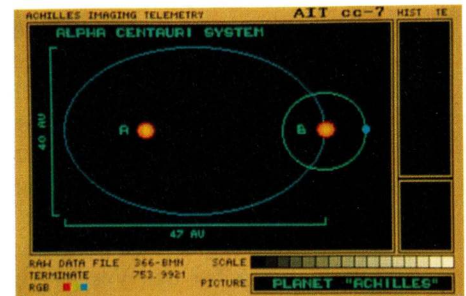
Computermalschule, Teil 3

Erforschung fremder Welten

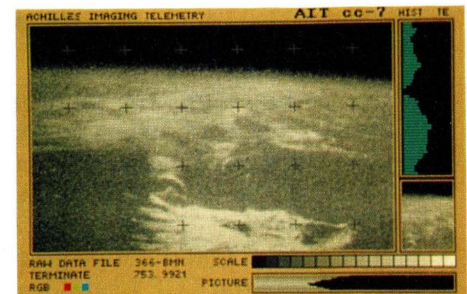
von Joel Hagen

Innenleben und Grafikeigenschaften des Amiga werden manchem genauso fremd erscheinen wie die Beschaffenheit des Alpha Centauri Systems. Der amerikanische Computerkünstler Joel Hagen schuf eine Computeranimation, die etwas von beidem vermittelt: Computergrafik und Astronomie. Spannend genug für eine 30-Minuten-Show im Fernsehen.

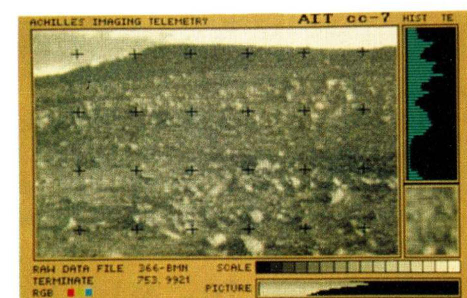
Als im Juni 1987 prominente Wissenschaftler, Künstler und Science-fiction-Autoren, zusammengeschlossen in einer Interessengemeischaft namens Contact, zu ihrem jährlichen Treffen in Los Angeles eintrafen, war klar, daß diese Tagung mit einem besonderen Ergebnis abschließen würde: eine grafische Dokumentation der Kongreßarbeit wurde dem lokalen Fernsehen präsentiert. Ich hatte bereits Erfahrung mit der grafischen Auswertung von Datenmaterial, das 1986 Voyager vom Uranus übermittelte; ebenso hatte ich DeluxPaint II und Digi-View zur Illustration von Science-fiction-Büchern eingesetzt. Mein Plan war, statt eines Zeichenbretts einen Amiga auf das Meeting mitzunehmen, um dessen Universalität zu demonstrieren. Neben akademischen Symposien und Workshops fand ein sehr interessantes Experiment statt. Eine Gruppe der Teilnehmer sollte Visionen über die Zukunft der Menschheit entwickeln, eine andere die Möglichkeit außerirdischen Lebens diskutieren. Aus der Arbeit der zweiten Gruppe entstand die Probe-Sequenz. Plastisches Modellieren, Fotografieren und Computermalen waren die Bestandteile meiner Arbeit, die schließlich zur besagten Animation führten. Aus dem Material der Contact-Gruppe modellierte ich einen kompletten Schädel und fotografierte ihn aus allen Winkeln. Diese Photos digitalisierte ich dann mit Digi-View (Grundlage für die Bilder der B-Sequence), ebenso wie Landschaftsfotos, die ich in Basaltgebieten auf Hawaii geschossen hatte, die für die Landschaft fremder Welten herhalten sollten (Sequenz A6 — A9). Mit DeluxePaint II wurden die Bilder bearbeitet und die Farbpaletten aufgeräumt (mit Color-Changer aus diesem Heft geht das sicherlich leichter — Anm. d. Red). Mit einer alten Schwarzweiß-Videokamera, die sich als sehr lichtempfindlich erwies, konnte ich herrliche Aufnahmen von meinen Modellen und Fotos machen, ohne im gleißenden Licht arbeiten zu müssen. An Digi-View ärgerte mich lediglich, daß es, um die Menüleisten leicht lesbar zu machen, die Farbpalette verfälschte. Um mit den Funktionen „shade" und „blend" in DeluxePaint II vernünftig arbeiten zu können, hilft nur eine Reorganisation der Farben mit der „spread"- Funktion.



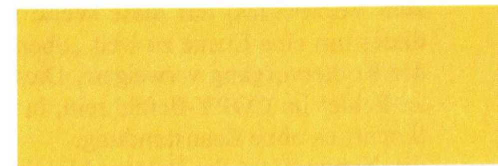
A 1: Position des Planeten Achilles im Alpha Centauri System.



A 5: Landmassen erheben sich aus dem Ozean.

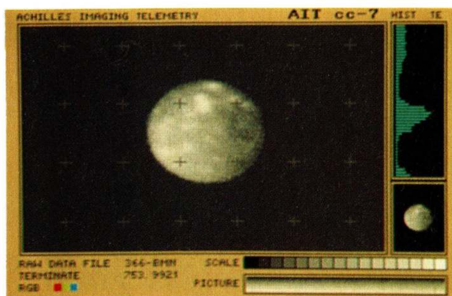


A 9: Unten rechts ist jetzt der Schädel zu erkennen.

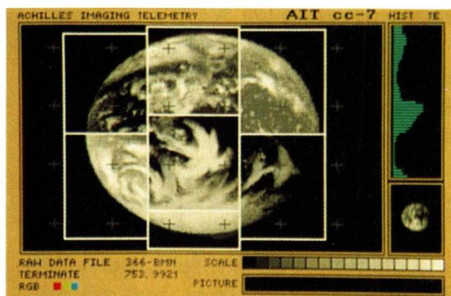


Startvorbereitungen zur interplanetarischen Erkundung

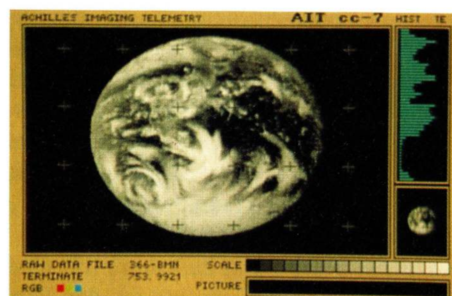
Mit den Disketten bewaffnet machte ich mich auf die Konferenz. Der Computerraum war vollgepropft mit Hardware, Kabeln, Videoanlagen und Bildschirmen — es sah aus wie in einem Flugkontrollzentrum. Schnell gesellten sich Computergrafiker dazu. Keith Doyle arbeitete an ▶



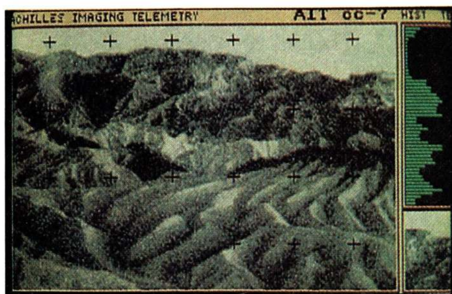
A 2: Erste bildfüllende Aufnahme des Planeten Achilles von der Sonde.



A 3: Aus zusammengesetzten Nahaufnahmen entsteht ein erster Eindruck der Oberfläche.



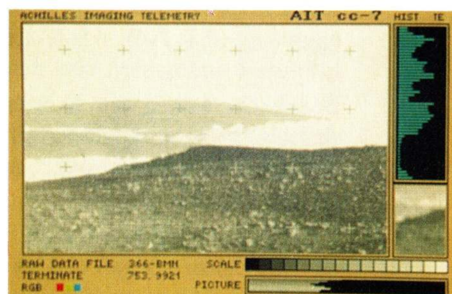
A 4: Je näher die Sonde dem Planeten kommt, um so detaillierter werden die Aufnahmen.



A 6: Kurz vor der Landung sind die Landschaftsformen deutlich zu erkennen.



A 7: Das weiße Rechteck markiert den Bereich, in dem später der Schädel gefunden wird.



A 8: Erstmals wird die steinige Bodenform deutlich.



A 10: Diese Nahaufnahme zeigt nun ganz deutlich den Schädel zwischen den Steinen.



A 11: Der Schädel soll nun grafisch hervorgehoben werden.



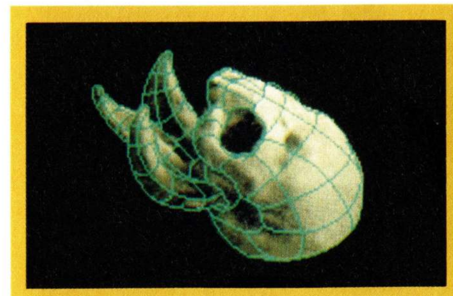
A 12: Jetzt ist der Hintergrund komplett entfernt worden.

einer Sprachsynthese, die die Animation des fremden Wesens ergänzen sollte. Darrel Anderson schuf tolle Animationen von Raumschiffen. In kurzer Zeit standen fünf Amigas, einige Speichererweiterungen, Festplatten, Drucker und nachleuchtende Farbmonitore herum. Am zweiten Tag geriet das Team richtig in Fahrt. Wir produzierten telemetrische Simulationen und Falschfarbenbilder von dem fremden Planeten, dem sich das „human team" näherte. Mit „range" produzierte ich nun eine Farbpalette von hellgelb bis dunkelgrün, füllte den Bildschirm mit der dunkelsten Farbe und definierte mir einen 2,5 cm großen, kreisförmigen Brush. Mit „shade" bewegte ich den Brush heftig über den Bildschirm, mal mit der linken, mal mit der rechten Maustaste gedrückt, um die Farben aufzuhellen oder abzudämpfen. Mit immer kleineren Brushes wiederholte ich den Vorgang. Nach einer Minute entstanden interessante Zufallslandschaften. Einige Details, wie Fjorde und Kanäle, wurden mit kleineren Brushes eingefügt (A3, A4). Als letztes füllte ich die grüne Flächen mit blau. Dies hob die Landmassen von den Wassergebieten ab.

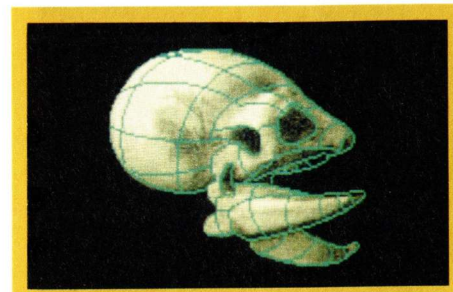
Begeistert waren die Konferenzteilnehmer vom Aegis Animator, dem damals faszinierendsten Animationsprogramm. Wir verwendeten eins der digitalisierten Modelle, um Kieferbewegungen in einer Endlosschleife zu simulieren.

Reichlich nervös war das Fernsehteam, als es uns bei der Arbeit zuschaute. Der Zeitdruck saß uns im Nacken. Ich hatte vor, eine realistische telemetrische Anlage zu simulieren. Annäherung an den Planeten, Erfassung von Details, Entdeckung des Schädels und Rekonstruktion des fremden Wesens: alles sollte überzeugend wirken. Dafür brauchte ich die volle Farbpalette, Page-Flipping innerhalb der Anzeigetafeln und Textdisplay. Aus Erfahrung wußte ich, daß keine der damals vorhandenen Software dazu in der Lage war. Keith Doyle jedoch hatte eine Art Animations-Programmiersprache entwickelt. Beim Vorbereitungsgespräch zeigte er sich überzeugt, daß sein Programm, The Director, meinen Ansprüchen gerecht würde. Die Beta-Version seines Programmes konnte mich schnell überzeugen. Mit einfachen, an BASIC angelehnten Befehlen, läßt sich alles steuern. Ich habe den Schädel in die Felslandschaft eingebaut (A8 — A10) und Keith bastelte eine Sequence, in der der Computer diesen Fund von der Umgebung freizustellen schien (A11 — A12). Mit der Kurvenfunktion von DPaint entwarf ich in der Zwischenzeit Drahtmodelle von dem Schädel (B1 — B5). Danach malte ich Muskelpartien über die Knochen, indem ich zuvor das Bild mit „fix background" schützte (B8). Dabei hat man das Gefühl, auf einer Glasplatte zu arbeiten, unter der die Vorlage liegt. Diese Bilder wurden in eine Sequenz eingebaut, in der sich das Drahtmodell dreht und dann als solider Schädel von allen Seiten betrachtet werden kann. Danach werden die Muskeln hinzugefügt und am Ende erscheint das vollständig rekonstruierte Antlitz des fremden Wesens. Als das Filmteam anrückte, konnten wir eine solide Arbeit vorführen. Dabei ließ sich die Sequenz direkt am Videoausgang des Amiga 1000 abnehmen. Die Reaktionen nach der Ausstrahlung der Animation waren umwerfend. Hat Amiga doch bewiesen, daß er auf diesem und jedem anderen Planeten der beste Desktop Video Homecomputer ist.

(Übersetzt von uk)



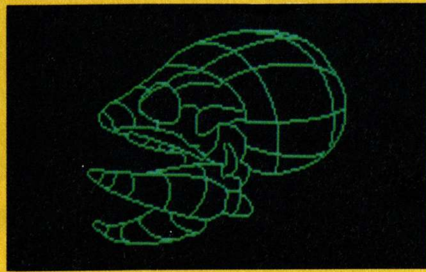
B 1: Über das Schädelbild wird ein Drahtmodell gelegt.



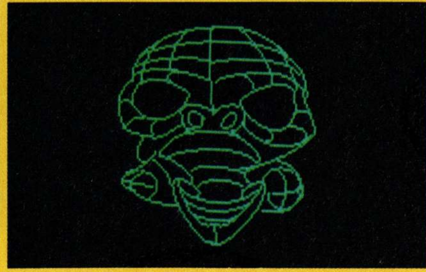
B 5: Nach Verschwinden des Drahtmodells . . . .



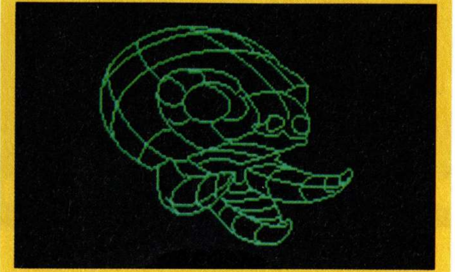
B 9: Schließlich wurde der Kopf des Alien ausmodelliert.



B 2: Mit der Kurvenfunktion von DPaint ist das Drahtmodell fertiggestellt . . . .



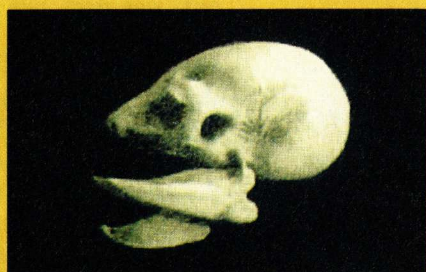
B 3: . . . und dreht sich, bis man eine Vorderansicht erhält.



B 4: Eine weitere Drehung offenbart die rechte Gesichtshälfte.



B 6: . . . wiederholt sich die Rotation des Schädels . . . .



B 7: . . . bis wieder die linke Seite zu sehen ist.



B 8: Über das fixierte Bild werden nun die Muskeln gemalt.



B 10: Farben geben dem Bild den letzten realistischen Touch.



B 11: Der rekonstruierte Kopf beginnt sich nun zu drehen . . . .



B 12: . . . bis man dem fremden Wesen von Achilles in die Augen schaut.

Wie IFF das Laufen lernte



Zeichnung: Hans Wernitz

Seit „Juggler", der mittlerweile legendären Ray-Tracing-Sequenz, ist der Begriff „Animation" mit dem Amiga nicht mehr wegzudenken. Heute unterstützen Programme, wie Aegis-VideoScape-3D oder Byte-by-Byte's „Animate-3D" den Anwender und Programmierer bei bewegten Bildern auf vielfältige Weise.

Gleich bei der Vorstellung des Amiga konnte Commodore mit einem Standard für den Datenaustausch aufwarten: IFF, eine Abkürzung von Interchange File Format. Dieses in Zusammenarbeit mit Electronic Arts entwickelte Dateiformat, inspiriert durch andere derartige Versuche (etwa DIF), sollte gleich zu Beginn den Fehler anderer Computermodelle verhindern, daß jeder Softwarehersteller sein eigenes Verfahren zur Ablage von Bildern, Texten, Musik oder anderer Daten konzipieren muß und so den Austausch von Daten zwischen verschiedenen Anwendungen unmöglich oder eine Unzahl an Konvertierungsprogrammen nötig macht.

Nach relativ kurzer Eingewöhnungszeit hat sich dieser Standard auch auf breiter Front durchgesetzt und ist heute nicht mehr wegzudenken. Neben den akzeptierten Formaten für Grafik (ILBM), Musik (SMUS) und Sounds (8SVX) fristet FTXT (Textformatierung) zur Zeit noch mehr ein kümmerliches Dasein, da sich Programme wie ProWrite und VizaWrite nicht an dieses Konzept halten und völlig eigene Verfahren zur Aufzeichnung des in diesen Programmen möglichen komplexen Seitenlayouts entwickelt haben, anstatt FTXT zu erweitern.

Anders wurden für Animationen bereits recht früh von Aegis die Spezifikationen für das neue Format, ANIM genannt, veröffentlicht. Es baut sinnvollerweise auf dem Verfahren zur Speicherung von unbewegten Bildern (ILBM) auf, was zeigt, wie flexibel IFF konzipiert wurde.

ILBM — Grundlage für Einzelbilder

Zur Verdeutlichung wird hier noch einmal kurz auf diejenigen Chunks von ILBM eingegangen, die auch bei ANIM von Bedeutung sind. Dabei wird vorausgesetzt, daß Ihnen der grundlegende Aufbau von IFF-Dateien bekannt ist. Informationen hierzu können Sie im Anhang B des ersten ROM Kernel Manuals (Exec) nachschlagen. Wollen Sie nur einen generellen Eindruck gewinnen, ohne sich für programmiertechnische Details zu interessieren, so empfehle ich Ihnen den Artikel „Electronic Arts setzt neuen Standard" im RUN-Sonderheft „ICON".

BMHD (BitMapHeaDer) — Der Bitmap-Vorspann

Dieses für einen ILBM-Chunk notwendige Attribut enthält die Informationen über die Größe der Grafik, die Anzahl der Bitplanes, das Seitenverhältnis eines Pixels, Komprimierungsverfahren und anderes mehr.

```
/* Art der Maskierung */

typedef UBYTE Masking;

#define mskNone                0
#define mskHasMask             1
#define mskHasTransparentColor 2
#define mskLasso               3

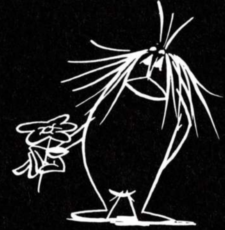
/* Datenkompressionsverfahren für BODY */

typedef UBYTE Compression;

#define cmpNone      0
#define cmpByteRun1 1

typedef struct
 {
 UWORD w, h;
       /* Breite und Höhe des Rasters */
 WORD  x, y;
       /* Pixelposition des Bildes */
```



```
UBYTE nPlanes;
        /* Anzahl der Bitplanes (Tiefe) */
Masking masking;
Compression compression;
UBYTE pad1;
        /* unbenutzt */
UWORD transparentColor;
        /* Nummer der transparenten Farbe */
UBYTE xAspect, yAspect;
        /* Seitenverhältnis eines Pixels */
WORD  pageWidth, pageHeight;
        /* Größe des Herkunftsrasters */
} BitMapHeader;
```

CMAP (ColorMAP) — Die Farbpalette

Die Größe der Farbpalette ist, von speziellen Amiga-Darstellungsmodi (siehe CAMG) abgesehen, meist 2^nPlanes.

```
typedef struct
 {
 UBYTE red, green, blue;
        /* Farbintensität von 0 bis 255 */
 } ColorRegister;

#define sizeofColorRegister 3
        /* gepackte Ablage! */
```

Die Farbpalette wird als Satz von Farbregistern abgelegt, wobei jedes Farbregister den Anteil der drei Grundfarben Rot, Grün und Blau angibt. Dieser Anteil bewegt sich im Bereich von 0 bis 255 und ist demnach für zukünftige (oder, bei entsprechendem Geldbeutel, durchaus auch gegenwärtige) Grafikhardware gerüstet, die mehr als 4096 Farben darzustellen vermag. Durch die gepackte Ablage der ColorRegister mit ungerader Länge mag einem CMAP-Chunk ein Füllbyte folgen.

CAMG (Commodore AMiGa) — Amiga Special

Der Amiga besitzt Darstellungsmodi, die die zu einem Pixel gehörigen Bits nicht auf die übliche Weise interpretieren. Dazu gehören insbesondere der Hold-And-Modify- (HAM) und Extra-Halfbrite-Modus. Um dem ILBM-Leseprogramm mitzuteilen, daß eine dieser speziellen Deutungen der Bitplanes gewünscht ist, wird dieser Chunk abgespeichert.

BODY — Die Planes

In diesem Chunk befinden sich die Daten für die Bitplanes und die Maske. Je nach Inhalt des Feldes „Compression" im BMHD sind diese Daten „roh" oder komprimiert abgelegt, beide Male in der Reihenfolge der Zeilen von „oben" nach „unten" und innerhalb der Zeilen von niederwertiger bis höherwertiger Plane (für die Bildung des Farbregisterindex) mit abschließender Maske (sofern vorhanden). Gepackt wird nur innerhalb von Zeilen, nicht über deren Grenzen hinaus.

Diese vier Attribute definieren vollständig ein unbewegtes Bild auf dem Amiga. Eine mögliche Weise der Ablage einer Animationssequenz stellt nun eine einfache Verkettung solcher Bilder dar. IFF böte hierfür die Chunk-Typen „CAT" und „LIST", wobei letzterer schon der Tatsache Rechnung trägt, daß die Bilder einer Animation gewisse gemeinsame Attribute (BMHD, CMAP, CAMG) besitzen, während „CAT" eine einfache Aneinanderfügung der Einzelbilder darstellte.

Beide Verfahren jedoch stellen eine grobe Verschwendung von Speicherplatz dar, da der besondere Effekt eines Films, im Unterschied zu einer Diashow, schließlich darin liegt, daß aufeinanderfolgende Bilder sich „ähnlich" sind, Änderungen von Einzelbild zu Einzelbild also nur bestimmte Regionen des Gesamtbildes betreffen.

ANIM — bringt Bewegung ins Bild

Im Unterschied zum allgemein anerkannten IFF-ILBM ringen im Bereich der Animation noch verschiedene Verfahren um die Vorherrschaft, erkennbar an verschiedenen Abspielprogrammen, die verwendet werden müssen, um eine Animation zum Leben zu erwecken.

Allen gemein ist, daß verschiedene Einzelbilder kombiniert und in Form einer einzigen großen Datei auf Diskette ge- ▶

speichert werden. Um nun die aus den oben genannten Verfahren resultierende Verschwendung von Diskettenplatz zu umgehen, wird lediglich das erste Bild einer Animation als FORM-ILBM gespeichert, von allen weiteren Bildern werden nur noch die Unterschiede, die Differenz zum direkt vorherigen (oder irgendeinem vorhergehenden — siehe unten) Bild abgelegt. Dies verkürzt die Animationsdatei beträchtlich, je nach verwendetem Kompressionsverfahren auf ein Viertel der Summe der Einzelbilder oder gar weniger.

Bei der Wahl der Kompressionsverfahren muß natürlich ein sehr wichtiger Punkt beachtet werden: Die Daten müssen sehr schnell dekomprimiert werden können, um ein flüssiges Abspielen zu gewährleisten! Andernfalls müßten alle Einzelbilder erst einmal entpackt werden, um dann abgespielt zu werden. Dies verlagert das Problem der Speicherplatzverschwendung von der Diskette in den Hauptspeicher des Rechners. Zur Abschätzung: Ein „rohes" HAM-Bild in NTSC-Auflösung benötigt rund 47 Kilobytes. Für PAL kommen 28 Prozent dazu, Interlace verdoppelt den Gesamtbetrag noch einmal.

Double-Buffering

Beim Abspielen bewegter Grafik wird üblicherweise mit zwei Bitmaps (und den jeweils zugehörigen Planes) gearbeitet: Eine, die gerade auf dem Bildschirm angezeigt wird, und eine weitere, in der das neue Einzelbild aufgebaut wird. Auf diese Weise wird Flackern, das entstünde, wenn eine Region des Bildspeichers gleichzeitig dargestellt und modifiziert würde, verhindert.

Nun bietet es sich an, das gerade zurückgenommene Bild als Grundlage für das folgende Bild zu verwenden. So verwendet IFF-ANIM üblicherweise auch ein Verfahren, das nicht auf den Unterschieden zweier „angrenzender" Einzelbilder der Animation beruht, sondern vielmehr die Unterschiede zum zwei Durchläufe zurückliegenden Bild nutzt. Sicherlich bestehen hier größere Unterschiede als zum direkt vorhergehenden Bild, so daß die Animationsdatei geringfügig länger wird, dafür ist aber ein schnelleres Abspielen möglich, da ein Kopiervorgang, nämlich vom aktuellen zum Puffer für das folgende Bild, wegfällt. Eine Ausnahme bilden das erste und zweite Bild, da ersteres, wie erwähnt, als normales ILBM gespeichert wird und für letzteres noch kein zwei Einzelbilder zurückliegendes Bild zur Verfügung steht.

Der allgemeine Aufbau einer ANIM-Datei ist wie folgt:

```
FORM ANIM
. FORM ILBM        Einzelbild #1
. . BMHD
. . CMAP
. . CAMG
. . BODY           reguläre ILBM-Daten
. FORM ILBM        Einzelbild #2
. . ANHD
. . DLTA           Unterschiede zu Einzelbild #1
. FORM ILBM        Einzelbild #3
. . ANHD
. . DLTA           Unterschiede zu Einzelbild #1
. FORM ILBM        Einzelbild #4
. . ANHD
. . DLTA           Unterschiede zu Einzelbild #2

usw.
```

Üblicherweise werden sich die im ersten FORM ILBM festgelegten Daten im Laufe der Animation nicht ändern, es ist aber durchaus nicht verboten. So kann es sinnvoll sein, die Farbpalette zu wechseln.

ANHD (ANimHeaDer) — Animationsvorspann

In obigem Schaubild sind zwei neue Chunktypen aufgeführt: ANHD und DLTA. Bei ANHD handelt es sich um ei-

```
typedef struct
 {
 UBYTE operation;
      /* verwendetes Kompressionsverfahren */
 UBYTE mask;
      /* nur XOR */
 UWORD w, h;
      /* nur XOR */
 WORD  x, y;
      /* nur XOR */
 ULONG abstime;
      /* Zeitdifferenz zum ersten
       Bild (60stel Sekunden) */
 ULONG reltime;
      /* Zeitdifferenz zum vorherigen
       Bild (60stel Sekunden) */
 UBYTE interleave;
      /* Um wieviele Bilder liegt
       die Differenzgrundlage zurück? */
 UBYTE pad[21];
      /* zur Zeit nicht verwendet */
 } AnimHeader;
```



nen Vorspann ähnlich BMHD. Er gibt an, wie die im DLTA-Chunk abgelegten Daten zu interpretieren sind, auf welches vorhergehende Bild sie sich beziehen und in welchem Zeitabstand die Einzelbilder gespielt werden sollen.
Für „operation“ sind vier verschiedene Werte definiert:
0: direkt gesetzt 1: XOR mode 2: Long Delta mode 3: Short Delta mode
„abstime“ und „interleave“ sind augenblicklich nicht verwendet. „interleave“ soll solche Anwendungen unterstützen, in denen die Unterschiede zum direkt vorhergehenden Bild als Grundlage für die Kompression dienen. Der Wert 0 entspricht dem Default von 2.

XOR — eXclusive OR

Der XOR-Modus ist mittlerweile als veraltet anzusehen, da die Delta-Modi wesentlich kürzere Animationsdateien erzeugen. Da ältere Programme aber möglicherweise noch immer diesen Modus unterstützen, hier der Vollständigkeit halber trotzdem die Spezifikationen:
Zwischen allen korrespondierenden Planes der beiden betroffenen Einzelbilder wird eine Exklusiv-Oder-Operation ausgeführt, so daß sich eine neue Bitmap ergibt, die genau an den Stellen eine „1“ enthält, an denen ein Unterschied zwischen den beiden Bildern besteht, ansonsten eine „0“. („Unterschied“ bezieht sich hier nicht notwendigerweise auf die Pixels, sondern auf die Bits der Planes.) Diese resultierenden Planes werden nun in Form eines BODY-Chunks mit normalem Byte-Run-Encoding (cmpByteRun1) abgelegt.
Die Grundlage für dieses Prinzip ist einfach: Stellen Sie sich zur Vereinfachung eine Grafik nur einer Plane Tiefe vor. Wird diese nun mit einem folgenden Bild durch XOR verknüpft, so entsteht eine Plane, die, bei entsprechend geringer Änderung des Bildinhalts, viele Nullen enthält. Nun werden durch cmpByteRun1 Strecken von Nullbytes (und andere) auf zwei Bytes reduziert, weshalb der XOR-BODY-Chunk wahrscheinlich kürzer als der des Einzelbildes sein wird.
Dieses Verfahren wird nun noch weiter verfeinert: So bestimmen die Elemente w, h, x und y des AnimHeaders das kleinste Rechteck, in dem sich alle Änderungen ergeben, die Maske „mask“ filtert diejenigen Planes aus, die unverändert bleiben, das entsprechende Bit ist dann gelöscht, ansonsten gesetzt.
Abgesehen von den verbesserten Kompressionsverfahren, liegt im XOR-Modus das Problem noch darin, daß die Berechnung der neuen Bitmap recht zeitaufwendig ist.
Modus 0 ist nicht näher spezifiziert, wird aber vermutlich ein komprimierter BODY-Chunk sein, der nicht der XOR-Operation unterworfen wurde. Dies mag sinnvoll sein, wenn ein „Schnitt“ in der Animation vorkommt.

DLTA (DeLTA) — Der Unterschied macht's

Der griechische Buchstabe Delta wird in Mathematik und Physik verwendet, um eine Differenz zu kennzeichnen. Gleichermaßen bezeichnet der DLTA-Chunk den Unterschied zwischen zwei Einzelbildern.
Die ersten acht Langworte eines solchen Chunks sind Byte-Zeiger in den Delta-Chunk, wobei dem ersten Byte des Chunks die Adresse 0 zukommt. Jedes dieser Langworte steht für eine der maximal acht Planes und zeigt auf den Bereich, der die Unterschiede näher beschreibt. Ist ein Zeiger gleich null, so werden in der entsprechenden Plane keine Modifikationen durchgeführt.
Programmtechnisch geht man wie folgt vor: Bei der Konstruktion der neuen Plane wird ein Zeiger auf den Beginn der Plane initialisiert.

Der DLTA-Bereich einer Plane (bezeichnet durch einen der acht Langwort-Zeiger) beginnt mit einem Offset, der in einem 16-Bit-Wort gespeichert ist. Ist dieser Offset positiv, so wird dieser Wert zum erwähnten Zeiger addiert, aber ... es gibt zwei verschiedene Operations-Modi: Short und Long Delta. Der Offset ist in der gewählten Größe zu interpretieren und entsprechend zu skalieren. Das folgende Wort (kurz oder lang, Sie wissen schon ...) wird dann an die aktuelle Position in der Plane gespeichert. Darauf folgt wieder ein Offset, der wieder ein Inkrement für den Planezeiger bezeichnet und darauf folgt wieder das eigentliche Datenwort. Ist der Offset gleich 0xFFFF oder -1, so ist das Ende der Differenzdaten dieser Plane erreicht.
Ist der Offset negativ, so bestimmt (Offset + 2) den eigentlichen Offset (schließlich ist es nicht sinnvoll, rückwärts zu gehen). Im Unterschied zu positiven Offsets folgt nun aber ein 16-Bit-Wort, das angibt, wieviele Datenworte vom gewählten Typ nun folgen. Dies eignet sich besonders dann, wenn mehrere aneinandergrenzende Datenworte sich ändern.

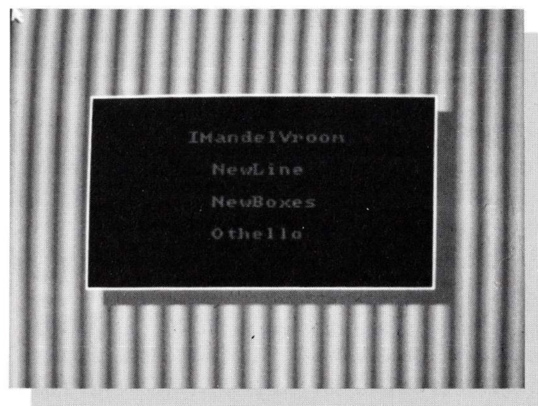
Der Short-Delta-Modus wurde zeitweilig vom Aegis-Videoscape-3D verwendet, Long-Delta soll auf 32-Bit-Amigas ein schnelleres Abspielen ermöglichen, wenngleich das resultierende Animationsfile meist etwas länger sein wird.
Die offizielle Dokumentation des von Aegis in Zusammenarbeit mit Sparta entwickelten ANIM-Standards definiert lediglich diese Compression-Codes von 0 bis 3. Spätere Versionen von Aegis-VideoScape-3D verwenden verbesserte „Packer“, aus denen kürzere Animationsdateien resultieren. Auch Byte-by-Byte bedient sich eines eigenen Kompressionsverfahrens, über das aber keine Einzelheiten verfügbar sind.
So Aegis will, können Sie zu den Verbesserungen des ANIM-Standards im nächsten Heft mehr lesen.

(Ralph Babel)

Bequemer Zugriff auf oft benutzte Programme

Häufig benutzte Software wird mit diesem AmigaWelt-Listing komfortabel. Ohne Umwege über Workbench oder CLI werden die wichtigsten Programme aufgelistet und lassen sich sofort mit der Maus starten.

Programmdisketten werden mit MakeCat übersichtlich. Es handelt sich bei diesem PRG um ein Utility, welches eine Menü-Oberfläche schafft. Diese erleichtert das Auswählen von unterschiedlichen PRGen einer Diskette durch Anklicken. Um ein Menü zu erstellen, ruft man MakeCat wie folgt auf:
Makecat [DF0:] [DF1:] [DF2:] [DF3:]
Wobei die nachfolgende Device-Adresse das Laufwerk angibt, auf der später das fertige Menü abgelegt werden soll. Nun erscheint eine Oberfläche mit zehn String-Gadgets. Hier kann man nun die PRG-Namen eintragen (mindestens zwei, maximal zehn), welche später im Menü erscheinen sollen. Hierbei ist darauf zu achten, daß alle Einträge nacheinander erfolgen und kein Gadget zwischen zwei PRG-Namen frei bleibt. Falls diese zentriert erscheinen sollen, so muß man auch den eingegebenen PRG-Namen zentrieren. Hat man alle Einträge vorgenommen, muß mit der Maus SAVE CATALOG angeklickt werden. Daraufhin wird das fertige Menü unter dem Namen CATALOG gespeichert. Falls Sie nun mehrere Disketten mit diesem Menü einrichten wollen, legen Sie einfach die entsprechende Diskette ins angewählte Laufwerk und wiederholen einfach das Anklicken der Taste. Andernfalls klicken Sie die Taste CLEAR CATALOG an und erstellen ein neues Menü oder beenden MakeCat durch Anklicken des Close-Gadgets. Nun müssen nur noch die PRG-Namen auf der Diskette in der Reihenfolge der Einträge im Menü „renamed" werden!
1. Eintrag = 1
2. Eintrag = 2
3. Eintrag = 3 etc.
Wenn der erste Eintrag „Perfect Sound" war, so wird dieser Name auf Diskette in „1" renamed! Danach muß man noch in die C-Dir den Befehl RUN kopieren und die Systemkonfiguration auf 80 Zeichen (was meistens der Fall ist) einstellen. Zum Schluß wird in die Startup-Sequence nur noch der Name CATALOG (welches jetzt ein eigenständiges PRG ist!) eingesetzt. Das wars. Wenn man nun die Diskette bootet, erscheint sofort das Menü auf dem Bildschirm. Man braucht nur noch den gewünschten PRG-Name anzuklicken, und das ausgewählte PRG wird geladen. Falls doch einmal Probleme beim Laden auftreten, sollte man das externe Laufwerk abschalten.
(Dirk Röman)



So sieht das Menü aus, das sofort nach dem Booten erscheint.

 Auf Leserdiskette



Eingabe mit Nanomon

Name: MakeCat **Sprache: Assembler**

```
0000 00 00 03 f3 00 00 00 00 00 00 00 02 00 00 00 00 1c
0010 00 00 00 01 00 00 04 01 00 00 00 01 00 00 03 e9 84
0020 00 00 04 01 42 43 53 40 67 00 04 40 0c 18 00 20 46
0030 66 00 00 0a 51 c8 ff f6 60 00 04 30 91 fc 00 00 1a
0040 00 01 02 90 df df ff ff 42 40 0c 90 44 46 30 3a f5
0050 67 00 00 2c 30 3c 00 01 0c 90 44 46 31 3a 67 00 9d
0060 00 1e 30 3c 00 02 0c 90 44 46 32 3a 67 00 00 10 37
0070 30 3c 00 03 0c 90 44 46 33 3a 66 00 03 ee 06 00 32
0080 00 30 13 c0 00 00 04 7e 42 80 2c 79 00 00 00 04 24
0090 43 f9 00 00 04 88 4e ae fe 68 23 c0 00 00 04 b8 26
00a0 43 f9 00 00 04 9a 4e ae fe 68 23 c0 00 00 04 c0 ad
00b0 43 f9 00 00 04 ab 4e ae fe 68 23 c0 00 00 04 bc 5f
00c0 2c 79 00 00 04 b8 4e ae fe da 23 c0 00 00 04 c4 68
00d0 20 40 20 10 04 80 00 00 00 2c 23 c0 00 00 04 ee 02
00e0 41 f9 00 00 04 d0 4e ae ff 34 23 c0 00 00 04 c8 56
00f0 20 40 23 e8 00 32 00 00 04 cc 2c 79 00 00 04 c0 0d
0100 30 3c 00 02 61 00 01 70 30 3c 00 02 32 3c 00 0a 47
0110 34 3c 00 ed 36 3c 00 86 61 00 03 44 30 3c 00 01 d0
0120 61 00 01 54 22 79 00 00 04 cc 42 69 00 24 33 7c be
0130 00 86 00 26 30 3c 00 ed 32 3c 00 86 4e ae ff 0a 82
0140 30 3c 00 77 32 3c 00 87 34 3c 00 79 36 3c 00 94 24
0150 61 00 03 0c 30 3c 00 03 61 00 01 1c 61 00 02 dc 0a
0160 61 00 02 ec 30 3c 00 03 61 00 01 18 30 3c 00 02 29
0170 4e ae fe aa 61 00 01 18 61 00 01 36 2c 79 00 00 67
0180 04 c0 42 40 61 00 00 f0 3e 3c 00 09 22 79 00 00 8a
0190 04 cc 30 3c 00 18 32 3c 00 14 34 3c 00 d7 36 3c 85
01a0 00 1b 48 e7 f0 40 4e ae fe ce 4c df 02 0f 06 41 d0
01b0 00 0b 06 43 00 0b 51 cf ff ea 08 39 00 06 00 bf f6
01c0 e0 01 66 00 00 54 2c 79 00 00 04 c0 20 79 00 00 1a
01d0 04 c8 32 28 00 0c 30 28 00 0e 0c 41 00 88 65 00 58
01e0 00 38 0c 41 00 93 62 00 00 30 0c 40 00 05 65 00 73
01f0 00 28 0c 40 00 75 62 00 00 08 42 44 60 00 00 d4 41
0200 0c 40 00 7d 65 00 00 12 0c 40 00 ee 62 00 00 0a 74
0210 38 3c 00 01 60 00 00 bc 2c 79 00 00 00 04 20 79 1a
0220 00 00 04 c8 20 68 00 56 4e ae fe 8c 4a 40 67 00 69
0230 ff 8a 20 40 20 28 00 14 0c 40 02 00 66 00 ff 7c 7d
0240 2c 79 00 00 04 b8 20 79 00 00 04 c8 4e ae ff b8 83
0250 2c 79 00 00 00 04 22 79 00 00 04 bc 4e ae fe 62 e3
0260 22 79 00 00 04 b8 4e ae fe 62 22 79 00 00 04 c0 2a
0270 4e ae fe 62 4e 75 22 79 00 00 04 cc 4e ae fe aa 33
0280 4e 75 22 79 00 00 04 cc 4e ae fe a4 4e 75 22 79 d4
0290 00 00 04 cc 33 7c 00 0b 00 24 33 7c 00 90 00 26 0a
02a0 41 f9 00 00 09 87 30 3c 00 0d 4e ae ff c4 4e 75 f5
02b0 22 79 00 00 04 cc 33 7c 00 80 00 24 33 7c 00 90 1e
02c0 00 26 41 f9 00 00 09 94 30 3c 00 0d 4e ae ff c4 37
02d0 4e 75 3a 3c 00 02 3c 3c 00 03 4a 44 66 00 00 0a 7f
02e0 61 00 01 12 60 00 00 06 61 00 01 26 61 00 01 3e 6a
02f0 3a 3c 00 03 3c 3c 00 02 4a 44 66 00 00 c6 61 00 c7
0300 00 f4 42 40 41 f9 00 00 08 74 43 f9 00 00 0d d2 46
```

```
0310 24 48 26 49 4a 10 67 00 00 30 4a 10 67 00 00 08 65
0320 12 d8 60 00 ff f6 d5 fc 00 00 00 19 d7 fc 00 00 3d
0330 00 18 20 4a 22 4b 06 40 00 01 0c 40 00 0a 67 00 c9
0340 00 10 4a 10 66 00 ff d4 0c 40 00 02 65 00 00 5c 69
0350 33 c0 00 00 0e dc 3c 3c 00 02 2c 79 00 00 04 bc 8c
0360 22 3c 00 00 04 7c 24 3c 00 00 03 ee 0c 46 00 02 38
0370 67 00 00 08 24 3c 00 00 03 ed 4e ae ff e2 4a 80 08
0380 66 00 00 0e 04 46 00 01 66 00 ff d6 60 00 00 1c bd
0390 22 00 2f 01 24 3c 00 00 09 a2 26 3c 00 00 06 5c e1
03a0 4e ae ff d0 22 1f 4e ae ff dc 41 f9 00 00 0d d2 e1
03b0 30 3c 00 3b 20 fc 20 20 20 20 51 c8 ff f8 60 00 75
03c0 fd fa 61 00 00 4c 38 3c 00 09 41 f9 00 00 08 74 82
03d0 43 f9 00 00 05 80 42 10 42 69 00 08 42 69 00 10 b5
03e0 d1 fc 00 00 00 19 d3 fc 00 00 00 50 51 cc ff e8 a2
03f0 60 00 fd 8a 30 05 61 00 fe 7e 61 00 00 3e 30 05 3f
0400 61 00 fe 80 30 06 61 00 fe 6e 61 00 fe 82 4e 75 86
0410 30 05 61 00 fe 62 61 00 00 36 30 05 61 00 fe 64 6a
0420 30 06 61 00 fe 52 61 00 fe 88 4e 75 08 39 00 06 6e
0430 00 bf e0 01 67 00 ff f6 4e 75 30 3c 00 03 32 3c cf
0440 00 88 34 3c 00 74 36 3c 00 93 60 00 00 12 30 3c 77
0450 00 7c 32 3c 00 88 34 3c 00 eb 36 3c 00 93 22 79 0a
0460 00 00 04 cc 4e ae fe ce 4e 75 2c 79 00 00 00 04 e6
0470 43 f9 00 00 04 ab 4e ae fe 68 2c 40 4e ae ff c4 9f
0480 22 00 24 3c 00 00 05 1a 26 3c 00 00 00 39 4e ae e0
0490 ff d0 22 4e 2c 79 00 00 00 04 4e ae fe 62 4e 75 d4
04a0 64 66 30 3a 43 41 54 41 4c 4f 47 00 69 6e 74 75 79
04b0 69 74 69 6f 6e 2e 6c 69 62 72 61 72 79 00 67 72 26
04c0 61 70 68 69 63 73 2e 6c 69 62 72 61 72 79 00 64 7b
04d0 6f 73 2e 6c 69 62 72 61 72 79 00 00 00 00 00 00 0a
04e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 fa
04f0 00 00 00 00 00 c8 00 14 00 f0 00 96 02 01 00 00 2c
0500 02 00 00 10 0a 00 00 00 05 54 00 00 00 00 00 00 7e
0510 05 00 00 00 00 00 00 00 00 00 00 96 00 32 02 80 a5
0520 00 c8 00 0f 20 43 41 54 41 4c 4f 47 20 43 52 45 ff
0530 41 54 4f 52 20 62 79 20 44 43 52 20 00 00 55 53 36
0540 45 20 54 48 45 20 46 4f 52 4d 41 54 20 3d 3e 20 97
0550 4d 41 4b 45 43 41 54 20 5b 44 46 30 3a 5d 20 5b 73
0560 44 46 31 3a 5d 20 5b 44 46 32 3a 5d 20 5b 44 46 8c
0570 33 3a 5d 20 21 0a 0d 00 00 00 05 a4 00 18 00 14 73
0580 00 c8 00 0a 00 00 00 02 00 04 00 00 00 00 00 00 1d
0590 00 00 00 00 00 00 00 00 00 00 00 05 80 00 00 4b
05a0 00 00 00 00 00 00 00 08 74 00 00 09 6e 00 00 19 47
05b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 9c
05c0 00 00 00 00 00 00 00 00 00 00 05 f4 00 18 00 1f f5
05d0 00 c8 00 0a 00 00 00 02 00 04 00 00 00 00 00 00 ee
05e0 00 00 00 00 00 00 00 00 00 00 00 05 d0 00 02 ee
05f0 00 00 00 00 00 00 00 08 8d 00 00 09 6e 00 00 19 51
0600 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 56
0610 00 00 00 00 00 00 00 00 00 00 06 44 00 18 00 2a 67
0620 00 c8 00 0a 00 00 00 02 00 04 00 00 00 00 00 00 a8
0630 00 00 00 00 00 00 00 00 00 00 00 06 20 00 02 0d
0640 00 00 00 00 00 00 00 08 a6 00 00 09 6e 00 00 19 44
0650 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 27
0660 00 00 00 00 00 00 00 00 00 00 06 94 00 18 00 35 93
0670 00 c8 00 0a 00 00 00 02 00 04 00 00 00 00 00 00 79
0680 00 00 00 00 00 00 00 00 00 00 00 06 70 00 02 ae
0690 00 00 00 00 00 00 00 08 bf 00 00 09 6e 00 00 19 4e
06a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 f8
06b0 00 00 00 00 00 00 00 00 00 00 06 e4 00 18 00 40 bf
06c0 00 c8 00 0a 00 00 00 02 00 04 00 00 00 00 00 00 4a
06d0 00 00 00 00 00 00 00 00 00 00 00 06 c0 00 02 4f
06e0 00 00 00 00 00 00 00 08 d8 00 00 09 6e 00 00 19 58
06f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 c9
0700 00 00 00 00 00 00 00 00 00 00 07 34 00 18 00 4b 31
0710 00 c8 00 0a 00 00 00 02 00 04 00 00 00 00 00 00 04
0720 00 00 00 00 00 00 00 00 00 00 00 07 10 00 02 6e
0730 00 00 00 00 00 00 00 08 f1 00 00 09 6e 00 00 19 4b
0740 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 83
0750 00 00 00 00 00 00 00 00 00 00 07 84 00 18 00 56 5d
0760 00 c8 00 0a 00 00 00 02 00 04 00 00 00 00 00 00 d5
0770 00 00 00 00 00 00 00 00 00 00 00 07 60 00 02 0f
0780 00 00 00 00 00 00 00 09 0a 00 00 09 6e 00 00 19 02
0790 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 54
07a0 00 00 00 00 00 00 00 00 00 00 07 d4 00 18 00 61 89
07b0 00 c8 00 0a 00 00 00 02 00 04 00 00 00 00 00 00 a6
07c0 00 00 00 00 00 00 00 00 00 00 00 07 b0 00 02 b0
07d0 00 00 00 00 00 00 00 09 23 00 00 09 6e 00 00 19 0c
07e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 25
07f0 00 00 00 00 00 00 00 00 00 00 08 24 00 18 00 6c 12
0800 00 c8 00 0a 00 00 00 02 00 04 00 00 00 00 00 00 60
0810 00 00 00 00 00 00 00 00 00 00 00 08 00 00 02 cf
0820 00 00 00 00 00 00 00 09 3c 00 00 09 6e 00 00 19 ff
0830 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 df
0840 00 00 00 00 00 00 00 00 00 00 00 00 18 00 77 8b
0850 00 c8 00 0a 00 00 00 02 00 04 00 00 00 00 00 00 31
0860 00 00 00 00 00 00 00 00 00 00 00 08 50 00 02 70
0870 00 00 00 00 00 00 00 09 55 00 00 09 6e 00 00 19 09
0880 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 b0
0890 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0d
08a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 6a
08b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 c7
08c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 24
08d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 81
08e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 de
08f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 3b
0900 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 81
0910 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 de
```

```
0920 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 3b
0930 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 98
0940 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 f5
0950 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 52
0960 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 af
0970 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0c
0980 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 69
0990 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 c6
09a0 00 00 00 00 00 00 00 00 00 00 00 53 41 56 45 20 62
09b0 43 41 54 41 4c 4f 47 20 43 4c 45 41 52 20 43 41 7a
09c0 54 41 4c 4f 47 00 00 00 03 f3 00 00 00 00 00 00 d2
09d0 00 02 00 00 00 00 00 00 00 01 00 00 01 51 00 00 b3
09e0 00 01 00 00 03 e9 00 00 01 51 2c 79 00 00 00 04 77
09f0 43 fa 03 94 4e ae fe 68 23 c0 00 00 04 02 43 f9 df
0a00 00 00 03 ad 4e ae fe 68 23 c0 00 00 04 06 43 fa da
0a10 03 64 4e ae fe 68 23 c0 00 00 03 fe 2c 40 41 fa 8b
0a20 03 84 4e ae ff 3a 23 c0 00 00 03 fa 06 80 00 00 b6
0a30 00 2c 23 c0 00 00 03 f2 06 80 00 00 00 28 23 c0 fa
0a40 00 00 03 f6 2c 79 00 00 04 02 20 79 00 00 03 f2 20
0a50 43 fa 03 6a 30 3c 00 10 4e ae ff 40 42 47 42 45 f0
0a60 42 40 3c 3c 00 08 30 06 61 00 02 de 0c 45 00 01 7f
0a70 66 00 00 10 04 46 00 01 0c 46 00 07 66 00 00 18 7d
0a80 42 45 06 46 00 01 0c 46 00 10 66 00 00 0a 3a 3c 8e
0a90 00 01 04 46 00 01 22 79 00 00 03 f6 42 69 00 26 5c
0aa0 30 07 33 40 00 24 32 3c 00 f9 4e ae ff 0a 06 47 9b
0ab0 00 01 0c 47 01 40 66 00 ff ae 30 3c 00 08 61 00 62
0ac0 02 88 22 79 00 00 03 f6 30 3c 00 3c 32 3c 00 46 2c
0ad0 34 3c 01 0e 36 3c 00 be 4e ae fe ce 30 3c 00 04 4e
0ae0 61 00 02 66 22 79 00 00 03 f6 30 3c 00 32 32 3c e9
0af0 00 3c 34 3c 01 04 36 3c 00 b4 4e ae fe ce 30 3c 74
0b00 00 00 61 00 02 44 22 79 00 00 03 f6 30 3c 00 33 5d
0b10 32 3c 00 3d 34 3c 01 03 36 3c 00 b3 4e ae fe ce 48
0b20 30 3c 00 05 61 00 02 22 30 3c 00 00 22 79 00 00 b6
0b30 03 f6 4e ae fe a4 3e 39 00 00 05 16 0c 47 00 07 c5
0b40 65 00 00 2a 33 fc 00 41 00 00 05 1c 33 fc 00 49 37
0b50 00 00 05 1e 33 fc 00 0d 00 00 05 14 0c 47 00 08 93
0b60 65 00 00 0a 33 fc 00 0b 00 00 05 14 3c 3c 00 0a 91
0b70 be 46 67 00 00 40 41 f9 00 00 05 20 04 46 00 01 17
0b80 be 46 66 00 00 06 60 00 00 10 d1 fc 00 00 00 04 88
0b90 0c 46 00 02 66 00 ff e6 30 18 32 10 91 79 00 00 0b
0ba0 05 18 d3 79 00 00 05 1a d3 79 00 00 05 1c d3 79 36
0bb0 00 00 05 1e 04 47 00 01 45 fa 02 24 61 00 01 96 6e
0bc0 d5 fc 00 00 00 18 20 4a 22 79 00 00 03 f6 30 3c c2
0bd0 00 18 4e ae ff c4 51 cf ff e4 2c 79 00 00 03 fe 8c
0be0 20 79 00 00 03 fa 42 40 32 3c fe d4 4e ae ff 5e 02
0bf0 2c 79 00 00 04 02 42 47 42 46 06 79 00 01 00 00 fa
0c00 05 16 20 79 00 00 03 fa 32 28 00 10 34 28 00 12 65
0c10 0c 42 00 32 65 00 00 56 0c 42 01 04 62 00 00 4e 2f
0c20 36 39 00 00 05 1c 38 39 00 00 05 1e 30 3c 00 01 57
0c30 b2 43 65 00 00 1e b2 44 62 00 00 18 be 40 67 00 c8
0c40 00 8a 4a 47 67 00 00 06 60 00 00 22 3e 00 60 00 84
0c50 00 2e d6 79 00 00 05 14 d8 79 00 00 05 14 06 40 eb
0c60 00 01 b0 79 00 00 05 16 66 00 ff c6 4a 47 67 00 b6
0c70 ff 92 3a 3c 00 05 3c 3c 00 01 60 00 00 06 3a 3c b4
0c80 00 04 22 79 00 00 03 f6 30 39 00 00 05 14 c0 c7 bd
0c90 d0 79 00 00 05 1a 33 40 00 26 33 7c 00 3c 00 24 b5
0ca0 22 79 00 00 03 f6 30 05 61 00 00 9e 41 f9 00 00 5c
0cb0 03 f4 20 3c 00 00 00 18 c0 c7 d1 c0 22 79 00 00 a9
0cc0 03 f6 30 3c 00 18 4e ae ff c4 0c 46 00 01 66 00 9d
0cd0 00 06 42 47 42 46 08 39 00 06 00 bf e0 01 66 00 e9
0ce0 ff 22 2c 79 00 00 03 fe 20 79 00 00 03 fa 4e ae f7
0cf0 ff be 2c 79 00 00 00 04 22 79 00 00 04 02 4e ae f6
0d00 fe 62 22 79 00 00 03 fe 4e ae fe 62 0c 47 00 0a a6
0d10 67 00 00 08 df 39 00 00 04 0a 2c 79 00 00 04 06 c6
0d20 4e ae ff ca 24 00 4e ae ff c4 26 00 22 3c 00 00 db
0d30 04 0a 4e ae ff 22 2c 79 00 00 00 04 22 79 00 00 2f
0d40 04 06 4e ae fe 62 4e 75 22 79 00 00 03 f6 4e ae 5e
0d50 fe aa 4e 75 22 79 00 00 03 f6 30 3c 00 3c 32 39 0c
0d60 00 00 05 18 3c 39 00 00 05 14 cc c7 92 46 4e ae 5d
0d70 ff 10 4e 75 69 6e 74 75 69 74 69 6f 6e 2e 6c 69 98
0d80 62 72 61 72 79 00 67 72 61 70 68 69 63 73 2e 6c a4
0d90 69 62 72 61 72 79 00 64 6f 73 2e 6c 69 62 72 61 c5
0da0 72 79 00 00 00 00 01 2c 01 40 00 fa 00 04 00 00 0e
0db0 00 00 00 0f 00 00 00 00 00 00 00 00 00 00 00 00 c3
0dc0 00 00 00 00 0a c0 00 70 00 02 00 05 00 08 00 09 9b
0dd0 00 0a 00 0b 00 0c 00 0d 00 0e 00 0f 00 00 00 00 e9
0de0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 7b
0df0 00 00 00 00 30 00 20 20 20 20 20 20 20 20 20 20 88
0e00 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 1e
0e10 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 7b
0e20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 d8
0e30 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 35
0e40 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 92
0e50 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 ef
0e60 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 4c
0e70 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 a9
0e80 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 06
0e90 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 63
0ea0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 c0
0eb0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 1d
0ec0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 7a
0ed0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 d7
0ee0 20 20 20 20 20 20 43 6f 70 79 72 69 67 68 74 20 2d
0ef0 62 79 20 44 43 52 20 49 49 2f 38 38 20 20 00 14 a2
0f00 00 02 00 aa 00 3c 00 49 00 51 00 06 00 05 00 0d e1
0f10 00 09 00 0e 00 05 00 00 ff f6 00 09 00 01 00 15 a8
0f20 00 09 00 21 00 11 00 28 00 1e 30 00 64 63 00 00 31
```

```
0f30 03 ec 00 00 00 35 00 00 00 00 00 00 00 10 00 00 36
0f40 00 16 00 00 00 20 00 00 00 2e 00 00 00 3e 00 00 3d
0f50 00 4a 00 00 00 56 00 00 00 5c 00 00 00 62 00 00 16
0f60 00 ae 00 00 00 da 00 00 00 fc 00 00 01 1e 00 00 2c
0f70 01 44 00 00 01 4e 00 00 01 5e 00 00 01 66 00 00 bc
0f80 01 6e 00 00 01 7e 00 00 01 8e 00 00 01 b4 00 00 91
0f90 01 ba 00 00 01 c0 00 00 01 c6 00 00 01 e0 00 00 b0
0fa0 01 f2 00 00 01 f8 00 00 02 08 00 00 02 14 00 00 7d
0fb0 02 1a 00 00 02 38 00 00 02 3e 00 00 02 6a 00 00 08
0fc0 02 70 00 00 02 7a 00 00 02 9a 00 00 02 a0 00 00 3f
0fd0 02 a8 00 00 02 b8 00 00 02 c4 00 00 02 d4 00 00 70
0fe0 02 fa 00 00 03 00 00 00 03 10 00 00 03 1a 00 00 28
0ff0 03 2c 00 00 03 32 00 00 03 44 00 00 03 54 00 00 0c
1000 03 60 00 00 03 6c 00 00 03 76 00 00 03 7c 00 00 7a
1010 00 00 00 00 03 f2 00 00 03 eb 00 00 00 01 00 00 39
1020 03 f2 00 00 00 00 00 00 00 00 03 ec 00 00 00 54 92
1030 00 00 00 00 00 00 00 60 00 00 00 6e 00 00 00 78 cd
1040 00 00 00 7e 00 00 00 88 00 00 00 8e 00 00 00 98 b0
1050 00 00 00 9e 00 00 00 a8 00 00 00 b8 00 00 00 be 3d
1060 00 00 00 c8 00 00 00 d2 00 00 00 d8 00 00 01 02 7f
1070 00 00 01 5a 00 00 01 6a 00 00 01 a4 00 00 01 aa a1
1080 00 00 01 fc 00 00 02 1e 00 00 02 24 00 00 02 34 55
1090 00 00 02 3e 00 00 02 48 00 00 02 54 00 00 02 60 17
10a0 00 00 02 6c 00 00 02 7e 00 00 02 8e 00 00 02 a0 52
10b0 00 00 02 e2 00 00 02 e8 00 00 03 2e 00 00 03 38 91
10c0 00 00 03 3e 00 00 03 72 00 00 03 88 00 00 03 a8 68
10d0 00 00 03 ae 00 00 04 3c 00 00 04 4e 00 00 04 60 34
10e0 00 00 04 e2 00 00 04 ea 00 00 05 54 00 00 05 76 d6
10f0 00 00 05 80 00 00 05 84 00 00 05 a4 00 00 05 c6 95
1100 00 00 05 d0 00 00 05 d4 00 00 05 f4 00 00 06 16 28
1110 00 00 06 20 00 00 06 24 00 00 06 44 00 00 06 66 cc
1120 00 00 06 70 00 00 06 74 00 00 06 94 00 00 06 b6 69
1130 00 00 06 c0 00 00 06 c4 00 00 06 e4 00 00 07 06 13
1140 00 00 07 10 00 00 07 14 00 00 07 34 00 00 07 56 b7
1150 00 00 07 60 00 00 07 64 00 00 07 84 00 00 07 a6 54
1160 00 00 07 b0 00 00 07 b4 00 00 07 d4 00 00 07 f6 f1
1170 00 00 08 00 00 00 08 04 00 00 08 46 00 00 08 50 ee
1180 00 00 08 54 00 00 00 00 00 00 03 f2 00 00 03 eb 68
1190 00 00 00 01 00 00 03 f2                         78
```

Nach Abtippen mit Nanomon sofort lauffähig: MakeCat.


Auf Leserdiskette



Überprüfung: mit checksum

```
  1 mousebot = $bfe001                                273
  2 execbase = $4                                     834
  3 openlib  = -408                                   801
  4 closelib = -414                                   501
  5 setApen  = -342                                   900
  6 setBpen  = -348                                   626
  7 ViewAdd  = -294                                   651
  8 recfill  = -306                                   103
  9                                                   103
 10 ;----------------------eingabestring  auf
     FORMAT ueberpruefen                              403
 11 clr d3                                            311
 12 subq #1,d0                                        686
 13 beq use_format                                    435
 14 ParLoop:                                          046
 15 cmp.b #$20,(a0)+                                  936
 16 bne ParFound                                      444
 17 dbra d0,ParLoop                                   422
 18 BRA Use_format                                    664
 19                                                   664
 20 ;----------------------                            765
 21 ParFound:                                         858
 22 sub.l #1,a0                                       779
 23 and.l #$dfdfffff,(a0)  ;in grosse letter
     umwandeln                                        077
 24 clr d0                                            414
 25 CheckDev:                                         633
 26 cmp.l #"DF0:",(a0)                                172
 27 beq start1                                        607
 28 move #1,d0                                        890
 29 cmp.l #"DF1:",(a0)                                994
 30 beq start1                                        429
 31 move #2,d0                                        565
 32 cmp.l #"DF2:",(a0)                                402
 33 beq start1                                        221
 34 move #3,d0                                        914
 35 cmp.l #"DF3:",(a0)                                132
 36 bne Use_format                                    334
 37 ;----------------------                            508
 38                                                   508
 39 start1:                                           833
 40 add.b #$30,d0             ; aktuelle device
     setzen                                           803
 41 move.b d0,catname+2                               256
 42 clr.l d0                                          491
 43 ;----------------------LIBS oeffnen                775
 44 move.l execbase,a6                                703
 45 lea intname,a1                                    791
 46 jsr openlib(a6)                                   427
 47 move.l d0,intbase                                 354
 48 lea graname,a1                                    749
 49 jsr openlib(a6)                                   217
 50 move.l d0,grabase                                 119
 51 lea dosname,a1                                    082
 52 jsr openlib(a6)                                   954
 53 move.l d0,dosbase                                 532
 54                                                   532
 55 ;----------------------screen handle
     setzen                                           054
 56 move.l intbase,a6                                 840
 57 jsr ViewAdd(a6)                                   759
 58 move.l d0,Viewbase                                842
 59 move.l d0,a0                                      820
 60 move.l (a0),d0                                    387
 61 sub.l #44,d0                                      226
 62 move.l d0,scrHandle                               211
 63 ;--------------------- window oeffen               830
 64 lea window_def,a0                                 235
 65 jsr -204(a6)                                      017
 66 move.l d0,winhandle                               932
 67 move.l d0,a0                                      878
 68 move.l 50(a0),rastport                            225
 69                                                   225
 70 ;--------------------- set color                   808
 71 move.l grabase,a6                                 213
 72 move #2,d0                                        157
 73 bsr do_Apen                                       334
 74 ;--------------------- main recfill oben           716
 75 move #2,d0                                        676
 76 move #10,d1                                       396
 77 move #237,d2                                      660
 78 move #134,d3                                      281
 79 bsr do_recfill                                    123
 80 ;--------------------- set apen                    971
 81 move #1,d0                                        302
 82 bsr do_Apen                                       595
 83 ;--------------------- line                        143
 84 move.l rastport,a1                                013
 85 clr 36(a1)                                        078
 86 move #134,38(a1)                                  226
 87 move #237,d0                                      776
 88 move #134,d1                                      307
 89 jsr -246(a6)                                      575
 90 ;--------------------- trennbox clear/save         240
 91 move #119,d0                                      609
 92 move #135,d1                                      873
 93 move #121,d2                                      013
 94 move #148,d3                                      143
 95 bsr do_recfill                                    385
 96 ;--------------------- set apen                    105
 97 move #3,d0                                        102
 98 bsr do_Apen                                       235
 99 bsr do_lfRec                                      664
100 bsr do_RiRec                                      338
101 ;--------------------- hintergrund color
     setzen                                           980
102 move #3,d0                                        641
103 bsr do_Bpen                                       535
104 ;-------------                                     063
105 move #2,d0                                        239
106 jsr setApen(a6)                                   234
107 ;--------------------- textausgebeb                757
108 bsr do_Lftxt                                      732
109 bsr do_ritxt                                      472
110 ;--------------------- set string box              424
111 start:                                            440
112 move.l grabase,a6                                 549
113 clr d0                                            670
114 bsr do_Apen                                       619
115 ;-----------------------                           505
116 move #9,d7                  ;counter              366
117                                                   366
118 move.l rastport,a1                                236
119 move #24,d0                                       088
120 move #20,d1                                       865
121 move #215,d2                                      909
122 move #27,d3                                       111
123 BoxLoop:                                          768
124 movem.l a1/d0-d3,-(a7)                            397
125 jsr RecFill(a6)                                   569
126 movem.l (a7)+,a1/d0-d3                            032
127 add #11,d1                                        907
128 add #11,d3                                        400
129 dbra d7,BoxLoop                                   527
130 ;------------------------------------              609
131 mainloop:                                         710
132 btst #6,mousebot                                  810
133 bne IDCMP_flag                                    685
134 ;--------------------- mouse_tst                   592
135 move.l grabase,a6                                 965
136 move.l winhandle,a0                               873
137 move 12(a0),d1              ; ypos                228
138 move 14(a0),d0              ; xpos                011
139 cmp #136,d1                                       141
140 blo IDCMP_Flag                                    144
141 cmp #147,d1                                       828
142 bhi idcmp_flag                                    933
143 ;--------------------- x position                  374
144 X1pos:                                            415
145 cmp #5,d0                                         195
146 blo idcmp_flag                                    022
147 cmp #117,d0                                       302
148 bhi X2pos                                         081
149 clr d4                                            110
150 bra InvBox                                        757
151                                                   757
152 X2pos:                                            991
153 cmp #125,d0                                       818
154 blo idcmp_flag                                    813
155 cmp #238,d0                                       029
156 bhi IDCMP_flag                                    574
157 move #1,d4                                        125
158 bra InvBox                                        444
159 ;--------------------- schliess gadget
     abfrage                                          483
160 IDCMp_Flag:                                       723
161 move.l execbase,a6                                875
162 move.l winhandle,a0                               623
163 move.l 86(a0),a0                                  709
164 jsr -372(a6)                                      445
165 tst d0                                            476
166 beq MainLoop                                      691
167 move.l d0,a0                                      973
168 move.l 20(a0),d0                                  654
169 cmp #$200,d0                                      639
170 bne mainloop                                      435
171 ;--------------------- prg ende                    945
172 END:                                              430
173 move.l intbase,a6                                 120
174 move.l winhandle,a0    ;close window              413
175 jsr -72(a6)                                       718
176                                                   718
177 ;--------------------- close LIBS                  746
178 move.l execbase,a6                                602
179 move.l dosbase,a1                                 138
180 jsr closelib(a6)                                  606
181 move.l intbase,a1                                 331
182 jsr closelib(a6)                                  839
183 move.l grabase,a1                                 803
184 jsr closelib(a6)                                  799
185 rts                                               676
186 ;------------------------------------------
     ---------------------                            689
187 ;--------------------- SUB ROT                     632
188 do_Apen:                                          316
189 move.l rastport,a1                                898
190 jsr setApen(a6)                                   457
191 rts                                               918
192 ;---------                                         002
193 do_Bpen:                                          895
194 move.l rastport,a1                                101
195 jsr setBpen(a6)                                   437
196 rts                                               034
197 ;---------                                         470
198 do_lftxt:                                         000
199 move.l rastport,a1                                998
200 move #11,36(a1)                                   122
201 move #144,38(a1)                                  023
202 lea txt1,a0                                       191
203 move #13,d0                                       969
204 jsr -60(a6)                                       107
205 rts                                               016
206 ;---------                                         436
207 do_ritxt:                                         951
208 move.l rastport,a1                                861
209 move #128,36(a1)                                  514
210 move #144,38(a1)                                  927
211 lea txt2,a0                                       148
212 move #13,d0                                       482
213 jsr -60(a6)                                       688
214 rts                                               521
215 ;-----------                                       540
216 InvBox:                                           668
217 move #2,d5                  ; box color
     invertieren                                      671
```

```
move #3,d6                                        722
tst d4              ; flag gesetzt (0 = links / 1 = rechts )      137
bne rt              ; flag = 1                    027
bsr InvLft                                        681
bra jp002                                         534
rt:                                               046
bsr InvRit                                        201
jp002:                                            943
bsr do_Leavebot                                   250
move #3,d5          ; orginal box color wieder erstellen          993
move #2,d6                                        271
tst d4              ; flag test                   095
bne rt1                                           391
;------------------------------------------ save rotiene         032
bsr InvLft                                        942
clr d0                                            183
lea strbuf_01,a0    ;stringbuffer                 656
lea saveprg+$430,a1 ;beginn der stringablage im CATALOG           839
                                                  839
move.l a0,a2        ;buf retten                   110
move.l a1,a3        ;catalogstart retten          963
                                                  963
tst.b (a0)          ;start strbuf =0?             803
beq FileMax2                                      091
;--------------------- stringbuf copieren !!      822
copyloop:                                         745
tst.b (a0)          ;ende strbuf ereicht ?        714
beq NextLine        ;ja dann nextline             479
move.b (a0)+,(a1)+                                225
bra copyloop                                      447
                                                  447
NextLine:                                         272
add.l #25,a2                                      780
add.l #24,a3                                      412
move.l a2,a0                                      629
move.l a3,a1                                      532
add #1,d0                                         797
cmp #10,d0                                        473
beq jp004                                         247
tst.b (a0)                                        204
bne copyloop                                      482
;---------------------- falls fileeintraege < 2 dann abbrechen   596
fileMax2:                                         546
cmp #2,d0                                         651
blo ENDsave                                       694
;----------------                                 449
jp004:                                            745
move d0,saveprg+$53a  ; anzahl der filenamen in CATALOG setzen !!!   761
;--------- open                                   811
move #2,d6          ; open durchlauf counter !    025
move.l dosbase,a6                                 026
openLoop:                                         012
move.l #CatName,d1                                266
move.l #1006,d2                                   179
cmp #2,d6           ; 1.ter durchlauf ?           968
beq jp005                                         659
move.l #1005,d2                                   971
jp005:                                            048
jsr -30(a6)                                       959
tst.l d0            ; d0 = 0 dann event. filename vorhanden       019
bne do_write                                      205
sub #1,d6           ; counter -1 und modus auf alt umstellen      619
bne openloop        ; und open noch einmal aufrufen !!!           129
bra ENDSAVE         ; falls wieder fehlerhaft dann abbrechen      209
;--------------------- write                      052
do_write:                                         619
move.l d0,d1                                      557
move.l d1,-(a7)                                   869
move.l #savePRG,d2                                783
move.l #1628,d3                                   303
jsr -48(a6)                                       087
;--------------------- close                      137
move.l (a7)+,d1                                   747
jsr -36(a6)                                       732
                                                  732
                                                  732
ENDSAVE:                                          000
lea savePrg+$430,a0                               985
move #59,d0                                       029
clrCatFile:                                       664
move.l #$20202020,(a0)+                           069
dbra d0,clrCatFile                                655
                                                  655
bra MainLoop                                      475
                                                  475
;--------------------- CLEAR                      670
rt1:                                              827
bsr InvRit                                        742
move #9,d4                                        269
lea strbuf_01,a0                                  210
lea stringinfo_01,a1                              045
clrLoop:                                          066
clr.b (a0)                                        409
clr 8(a1)           ; clr cursor pos              393
clr 16(a1)          ; clr sichtbare charact. im container !       159
add.l #25,a0                                      001
add.l #80,a1                                      809
dbra d4,clrloop                                   421
                                                  421
bra start                                         196
;---------------------- box links                 551
InvLft:                                           928
move d5,d0                                        680
bsr do_apen                                       309
bsr do_LfRec                                      442
move d5,d0                                        762
bsr do_Bpen                                       524
move d6,d0                                        521
bsr do_Apen                                       170
bsr do_Lftxt                                      513
rts                                               070
;---------------------- box rechts                181
                                                  181
InvRit:                                           287
move d5,d0                                        583
bsr do_apen                                       544
bsr do_RiRec                                      578
move d5,d0                                        474
bsr do_Bpen                                       644
move d6,d0                                        065
bsr do_Apen                                       578
bsr do_Ritxt                                      830
rts                                               199
                                                  199
;--------------------- SAVE/CLEAR felder setzen   907
do_LeaveBot:                                      457
btst #6,mousebot                                  725
beq do_LeaveBot                                   529
rts                                               406
;----------------                                 801
do_lfRec:                                         333
move #3,d0                                        898
move #136,d1                                      183
move #116,d2                                      059
move #147,d3                                      480
bra do_recfill                                    012
do_RiRec:                                         093
move #124,d0                                      638
move #136,d1                                      243
move #235,d2                                      697
move #147,d3                                      254
;---------                                        658
do_recfill:                                       211
move.l rastport,a1                                473
jsr RecFill(a6)                                   949
rts                                               498
;------------------                               035
use_format:                                       390
move.l execbase,a6                                574
lea dosname,a1                                    281
jsr -408(a6)                                      357
move.l d0,a6                                      765
jsr -60(a6)                                       559
move.l d0,d1                                      201
move.l #uf_txt,d2                                 001
move.l #uf_txtend-uf_txt,d3                       884
jsr -48(a6)                                       952
move.l a6,a1                                      526
move.l $4,a6                                      722
jsr -414(a6)                                      895
rts                                               644
                                                  644
;----------------------- PARAMETER                539
catName: dc.b "df0:CATALOG",0                     976
intname: dc.b "intuition.library",0               715
graname: dc.b "graphics.library",0                255
dosname: dc.b "dos.library",0                     280
even                                              238
intbase: dc.l 0                                   035
dosbase: dc.l 0                                   803
grabase: dc.l 0                                   843
viewbase: dc.l 0                                  748
winHandle: dc.l 0                                 589
rastport: dc.l 0                                  123
;------------------- WINDOW DEF                   220
window_def:                                       028
dc.w 200            ;x-pos                        120
dc.w 20             ;y-pos                        861
dc.w 240            ;bright                       950
dc.w 150            ;high                         427
dc.b 2              ;text-pen                     884
dc.b 1              ;ground-pen                   318
dc.l $200           ;IDCMP-flag                   484
dc.l $100a          ;active-flag                  885
dc.l gadget1        ;first-gad                    511
dc.l 0              ;checkMark                    336
dc.l Wtxt                                         096
;                                                 307
SCRhandle: dc.l 0                                 170
;                                                 221
dc.l 0              ;bitmap                       770
dc.w 150            ;minbright                    403
dc.w 50             ;minHigh                      023
dc.w 640            ;maxBright                    446
dc.w 200            ;maxHigh                      257
dc.w 15             ;screen-type                  973
even                                              091
                                                  091
wtxt: dc.b " CATALOG CREATOR by DCR ",0           662
even                                              844
UF_txt: dc.b "USE THE FORMAT => MAKECAT [DF0:] [DF1:] "   436
dc.b "[DF2:] [DF3:] !",$0a,$0d                    010
UF_txtend:                                        441
even                                              799
;##########---------- STRING GADGETS DEF 1 - 10   351
gadget1:                                          210
dc.l gadget2        ;weiteres gadgets             371
dc.w 24,20          ;x,y-pos des containers       272
dc.w 200,10         ;breite,hoehe                 898
dc.w 0                                            382
dc.w 2                                            180
dc.w 4              ;type StringGadgets           047
dc.l 0                                            320
dc.l 0,0,0                                        105
dc.l stringinfo_01                                569
dc.w 0                                            469
dc.l 0                                            590
;------                                           155
even                                              777
stringinfo_01:                                    454
dc.l strbuf_01                                    949
dc.l undo                                         024
CP01:dc.w 0                                       414
dc.w 25             ;cursor-pos,anzahl der max. zeichen   148
dc.w 0,0,0,0,0,0                                  604
dc.l 0,0,0                                        869
even                                              395
                                                  395
;##########-----------------------------          701
gadget2:                                          485
dc.l gadget3                                      133
dc.w 24,31                                        523
dc.w 200,10                                       974
dc.w 0                                            106
dc.w 2                                            320
dc.w 4                                            976
dc.l 0                                            689
dc.l 0,0,0                                        778
dc.l stringinfo_02                                219
dc.w 2                                            233
dc.l 0                                            938
;------                                           511
even                                              173
stringinfo_02:                                    447
dc.l strbuf_02                                    711
dc.l undo                                         322
CP2:dc.w 0                                        433
dc.w 25                                           344
dc.w 0,0,0,0,0,0                                  056
dc.l 0,0,0                                        289
even                                              239
                                                  239
;##########-----------------------------          161
gadget3:                                          110
dc.l gadget4                                      551
dc.w 24,42                                        051
dc.w 200,10                                       534
dc.w 0                                            626
dc.w 2                                            264
dc.w 4                                            856
dc.l 0                                            017
dc.l 0,0,0                                        874
dc.l stringinfo_03                                060
dc.w 2                                            026
dc.l 0                                            043
;------                                           972
even                                              202
stringinfo_03:                                    865
dc.l strbuf_03                                    770
dc.l undo                                         057
dc.w 0                                            957
dc.w 25                                           044
dc.w 0,0,0,0,0,0                                  036
dc.l 0,0,0                                        341
                                                  341
even                                              611
;##########-----------------------------          661
gadget4:                                          239
dc.l gadget5                                      113
dc.w 24,53                                        603
dc.w 200,10                                       070
dc.w 0                                            802
dc.w 2                                            056
dc.w 4                                            288
dc.l 0                                            385
dc.l 0,0,0                                        114
dc.l stringinfo_04                                109
dc.w 2                                            339
dc.l 0                                            092
;------                                           569
even                                              927
stringinfo_04:                                    763
dc.l strbuf_04                                    677
dc.l undo                                         072
dc.w 0                                            148
dc.w 25                                           463
dc.w 0,0,0,0,0,0                                  623
dc.l 0,0,0                                        336
even                                              166
;##########-----------------------------          696
gadget5:                                          359
dc.l gadget6                                      874
dc.w 24,64                                        258
dc.w 200,10                                       497
dc.w 0                                            437
dc.w 2                                            627
dc.w 4                                            939
dc.l 0                                            500
dc.l 0,0,0                                        445
dc.l stringinfo_05                                969
dc.w 2                                            743
dc.l 0                                            056
;------                                           005
```

```
even                                         043
stringinfo_05:                               436
dc.l strbuf_05                               631
dc.l undo                                    818
dc.w 0                                       966
dc.w 25                                      649
dc.w 0,0,0,0,0,0                             425
dc.l 0,0,0                                   962
even                                         312
;##########---------------------------       674
gadget6:                                     694
dc.l gadget7                                 522
dc.w 24,75                                   152
dc.w 200,10                                  487
dc.w 0                                       659
dc.w 2                                       713
dc.w 4                                       489
dc.l 0                                       194
dc.l 0,0,0                                   867
dc.l stringinfo_06                           968
dc.w 2                                       022
dc.l 0                                       303
;------                                      632
even                                         078
stringinfo_06:                               644
dc.l strbuf_06                               712
dc.l undo                                    751
dc.w 0                                       347
dc.w 25                                      338
dc.w 0,0,0,0,0,0                             666
dc.l 0,0,0                                   531
even                                         001
;##########---------------------------       755
gadget7:                                     284
dc.l gadget8                                 121
dc.w 24,86                                   205
dc.w 200,10                                  992
dc.w 0                                       364
dc.w 2                                       922
dc.w 4                                       730
dc.l 0                                       211
dc.l 0,0,0                                   468
dc.l stringinfo_07                           698
dc.w 2                                       248
dc.l 0                                       129
;------                                      978
even                                         584
stringinfo_07:                               379
dc.l strbuf_07                               272
dc.l undo                                    407
dc.w 0                                       195
dc.w 25                                      370
dc.w 0,0,0,0,0,0                             698
dc.l 0,0,0                                   819
even                                         801
;##########---------------------------       531
gadget8:                                     097
dc.l gadget9                                 071
dc.w 24,97                                   945
dc.w 200,10                                  276
dc.w 0                                       024
dc.w 2                                       510
dc.w 4                                       286
dc.l 0                                       039
dc.l 0,0,0                                   688
dc.l stringinfo_08                           759
dc.w 2                                       781
dc.l 0                                       966
;------                                      267
even                                         633
stringinfo_08:                               017
dc.l strbuf_08                               079
dc.l undo                                    578
dc.w 0                                       574
dc.w 25                                      017
dc.w 0,0,0,0,0,0                             305
dc.l 0,0,0                                   418
even                                         256
;##########---------------------------       850
gadget9:                                     397
dc.l gadget10                                679
dc.w 24,108                                  826
dc.w 200,10                                  881
dc.w 0                                       845
dc.w 2                                       331
dc.w 4                                       475
dc.l 0                                       676
dc.l 0,0,0                                   005
dc.l stringinfo_09                           853
dc.w 2                                       763
dc.l 0                                       708
;------                                      297
even                                         399
stringinfo_09:                               940
dc.l strbuf_09                               411
dc.l undo                                    646
dc.w 0                                       866
dc.w 25                                      813
dc.w 0,0,0,0,0,0                             845
dc.l 0,0,0                                   390
even                                         316
;##########---------------------------       270
gadget10:                                    389
dc.l 0                                       126
dc.w 24,119                                  483
dc.w 200,10                                  902
dc.w 0                                       122
dc.w 2                                       184
dc.w 4                                       392
dc.l 0                                       145
dc.l 0,0,0                                   026
dc.l stringinfo_10                           094
dc.w 2                                       148
dc.l 0                                       061
;------                                      230
even                                         644
stringinfo_10:                               165
dc.l strbuf_10                               456
dc.l undo                                    519
dc.w 0                                       435
dc.w 25                                      034
dc.w 0,0,0,0,0,0                             386
dc.l 0,0,0                                   547
even                                         273
;------------------------ STRINGINFO          372
;----------------------                      098
strBuf_01: blk.b 25,0                        674
strBuf_02: blk.b 25,0                        427
strBuf_03: blk.b 25,0                        225
strBuf_04: blk.b 25,0                        152
strBuf_05: blk.b 25,0                        132
strBuf_06: blk.b 25,0                        497
strBuf_07: blk.b 25,0                        147
strBuf_08: blk.b 25,0                        390
strBuf_09: blk.b 25,0                        854
strBuf_10: blk.b 25,0                        658
                                             658
undo: blk.b 25,0                             294
txt1: dc.b "SAVE CATALOG "                   438
txt2: dc.b "CLEAR CATALOG"                   615
even                                         917
SavePRG:                                     935
blk.b 1628,0                                 653
```

Assembler-Code zu MakeCat

Hintergründiges mit RunBack

Davon, daß der Amiga fähig zum Multitasking ist, profitiert auch, wenn nicht gar in erster Linie, der Programmierer: viele Aufgaben lassen sich elegant in Form separater Prozesse lösen, man denke hierbei an Programme wie „FACC“, den Floppy-Beschleuniger oder auch an den Virus-Detektor aus dem letzten Heft. Im Unterschied etwa zu GOMF2.0 beachten diese Programme jedoch gar nicht, ob Sie nun im Hintergrund laufen oder ohne RUN gestartet wurden. Und ebensowenig beachtet RUN die Eigenarten der durch ihn gestarteten Programme. Ihnen ist sicherlich bekannt, daß der letzte Befehl der „Startup-Sequence“, der Datei, die die Vorgänge beim Einschalten und nach dem Reset steuert, oft lautet . . .

```
EndCLI >NIL:
```

Dieser Befehl beendet bekanntlich die Existenz eines CLI-Prozesses und -Fensters. Daß die Ausgabe durch „>NIL:“ ins Nichts geleitet wird, hat lediglich die Bedeutung, die Abschiedsmeldung „CLI Task ending“ zu unterdrücken.
Haben Sie jedoch zuvor Programme durch RUN von diesem CLI aus gestartet, so wird das Fenster weiterhin bestehen bleiben und sich erst dann schließen lassen, wenn die Ausführung aller durch RUN heraufbeschwörten Programme ebenfalls beendet wurde.
Woher kommt dieses auf den ersten Blick doch etwas merkwürdige Verhalten? Geben Sie hierzu einmal ein . . .

```
Run Type S:Startup-Sequence
```

Sie werden bemerken, daß, obwohl durch „RUN“ ein eigenes „BackGround-CLI“ erzeugt wird, die Ausgabe noch immer in das CLI-Fenster erfolgt, von dem aus dieses Hintergrund-CLI ins Leben gerufen wurde.
Schließen Sie nun dieses Vordergrund-CLI, ohne daß der Hintergrund-Prozeß seine Ausgabe beendet hat, so verzögert AmigaDOS entsprechend das Schließen des zugehörigen Fensters. Intern wird dies wie folgt gehandhabt: Jedes Fenster besitzt einen Zähler, der angibt, wieviele Zugriffsberechtigungen auf dieses Fenster ausgegeben wurden, und erst, wenn dieser Zähler auf null zurückgegangen ist, wird das Fenster geschlossen.
Dieses Verhalten ist für alle durch „RUN“ gestarteten Programme der Fall, ganz gleich, ob diese überhaupt eine Ausgabe durch „printf()“ und artverwandte Funktionen vornehmen könnten oder nicht. Ein Programm wie „ViCheck“, das seine Meldungen dem Benutzer ja durch Auto-Requester bekanntgibt, wird demnach nicht anders behandelt als „Type“ im obigen Beispiel — sie alle erhöhen den internen Zähler! Dies ist jedoch unerwünscht, denn viele Benutzer wollen in ihrer Startup-Sequence „residente“ Programme durch „RUN“ starten und trotzdem das sogenannte „Initial CLI“, das CLI also, das direkt nach dem Booten sichtbar wird, schließen, um danach auf der Workbench zu arbeiten.

Dieses Problem wurde zuerst von Robert Peck, bekannt durch die Dokumentation zum Amiga, unzählige Beispielprogramme und auch Artikel in der amerikanischen AmigaWorld, gelöst: Sein „RunBackGround" (siehe auch Fish-Disk 73) erlaubte es erstmals, nach Start eines Programms das eigentlich nicht mehr zugehörige CLI-Fenster zu schließen. Allerdings gab es auch Probleme: Ein Absturz war die Folge, wenn das spezifizierte Programm gar kein Executable war. Auch gingen, selbst wenn der Background-Task jemals seine Ausführung beendete, bei jedem Aufruf von RunBackGround 40 Bytes an Speicher verloren. Sicherlich nicht tragisch, aber eine scheinbar nicht zu umgehende Restriktion, die aus der Art und Weise, wie Robert Peck das Problem löste, resultierte.

Charles Heath, den meisten wohl durch seinen „TxEd" und das „ARP" (AmigaDOS Replacement Project) bekannt, beseitigte zumindest einen Teil dieser Schönheitsfehler, indem er vor der Ausführung einige Tests durchführte, ob das spezifizierte Programm überhaupt existierte. Auch war seine Version, wie nicht anders zu erwarten, in Assembler geschrieben, so daß diese anstelle von 3268 Bytes nur noch 948 Bytes lang war. Sie findet sich zusammen mit dem Quelltext auf Fish-Disk 65. Das neue RunBack ist nur noch 340 Bytes lang und eliminiert zum Ausgleich für die geringe Größe auch die restlichen Probleme: Kein Speicher geht verloren und eine Absturzgefahr, die von RunBack ausgeht, besteht auch nicht mehr. Das hierdurch ins Leben gerufene Hintergrund-CLI besitzt keinen Console-Task und alle Kanäle zur Standard-Ein- und -Ausgabe zeigen ins Nichts, nach NIL: also. Jegliches I/O, das nicht eigene Files eröffnet, bleibt daher wirkungslos! Auch ist es nicht möglich, mittels „*" das aktuelle Process-Window anzusprechen, denn ein solches existiert schließlich gar nicht. Das gleiche gilt für Programmanweisungen wie . . .

```
Open("*", MODE_{OLD|NEW}FILE)
```

Sofern notwendig, kann ein eigenes Fenster in Form von . . .

RunBack C:Copy S:Startup-Sequence CON:0/0/640/256/

. . . spezifiziert werden. I/O-Umlenkung durch „>" und „<" ist — im Gegensatz zu den vorherigen Versionen von RunBack — durchaus möglich, auch bestehen keinerlei Begrenzungen für die Länge der Kommandozeile, da diese intern nicht weiterbearbeitet wird.

Übrigens hat ein durch RUN gestartetes CLI normalerweise lediglich Schreibzugriff auf das Original-CLI-Fenster.

Wenn Sie sich einmal ganz besonders elegant in eine Sackgasse manövrieren wollen, so starten Sie den Befehl „DiskCopy" doch einmal durch RUN — das Programm schaltet den Zugriff auf die spezifizierten Laufwerke mittels der Packet-Action „Inhibit" ab und wartet darauf, daß Sie RETURN drücken. Letzteres gestaltet sich bei einem Hintergrund-CLI nun aus genannten Gründen extrem schwierig, da eine Eingabe nicht definiert ist. Sollten Sie nicht ein weiteres CLI-Fenster oder die Workbench und dazu ein weiteres Laufwerk oder die RAM-Disk entsprechend bestückt haben, so ist alles zu spät. Ein Programm sollte auch gegen diese Art von Fehlbedienung gesichert sein, untersuchen Sie Ihre eigenen Werke einmal darauf. Um zu verhindern, daß zwei Prozesse (Initial-CLI, das die Startup-Sequence bearbeitet, und der durch RunBack kreierte) gleichzeitig auf eine Diskette zugreifen und dadurch die Lebensdauer sowohl von Diskette als auch Laufwerk nicht gerade verlängern (rrrrt, rrrrt, rrrrt, rrrrt, . . .), empfiehlt sich die Angabe eines WAIT-Kommandos nach RunBack. Sollten Sie einen RESIDENT-Befehl besitzen, so ist es natürlich noch besser, diesen vor RunBack auf WAIT anzuwenden. Bei der Spezifizierung des auszuführenden Kommandos sollte übrigens darauf geachtet werden, daß dem neuen Hintergrund-Prozeß weder das aktuelle Verzeichnis (CD), noch der Pfad (PATH) vererbt werden. Das Current Directory liegt, ähnlich dem Initial-CLI, auf „DF0:", ist jedoch nicht speziell auf die Diskette, die zur Zeit darin liegt, festgelegt. Kommandos im Directory C: werden natürlich auch ohne die Angabe des vollständigen Pfadnamens erkannt und ausgeführt.

(Ralph Babel)

Programmname: RunBack
Sprache: Assembler, Metcomco


Auf Leserdiskette



Eingabe mit Nanomon

```
000 00 00 03 f3 00 00 00 00 00 00 00 01 00 00 00 00 8b
010 00 00 00 00 00 00 00 4c 00 00 03 e9 00 00 00 4c 85
020 26 48 2c 78 00 04 93 c9 4e ae fe da 28 40 4a ac 92
030 00 ac 66 1c 41 ec 00 5c 4e ae fe 80 41 ec 00 5c f1
040 4e ae fe 8c 24 00 4e ae ff 7c 22 42 4e ee fe 86 c5
050 7e 14 43 fa 00 ea 70 21 4e ae fd d8 4a 80 67 72 fb
060 2a 40 7e 0c 20 3c 00 00 01 61 72 01 4e ae ff 3a b8
070 26 00 67 58 24 40 7e 0b 41 fa 00 74 22 40 70 60 4e
080 12 d8 51 c8 ff fc 70 0a 60 02 12 db b0 13 66 fa 5a
090 42 11 2c 2c 00 a4 42 ac 00 a4 41 fa 00 3a 22 08 31
0a0 74 00 14 2c 00 09 e4 8b 53 83 28 3c 00 00 0f a0 77
0b0 4e ad ff 76 29 46 00 a4 4a 80 67 04 7e 00 60 0c fd
0c0 22 4a 20 3c 00 00 01 61 4e ae ff 2e 22 4d 4e ae 82
0d0 fe 62 20 07 4e 75 52 61 6c 70 68 20 42 61 62 65 6c
0e0 6c 27 73 20 22 52 75 6e 42 61 63 6b 22 00 2c 78 e6
0f0 00 04 43 fa 00 4a 70 21 4e ae fd d8 4a 80 67 30 b5
100 2a 40 41 fa 00 46 22 08 24 3c 00 00 03 ee 4e ad ea
110 ff e2 4a 80 67 14 41 fa 00 37 22 08 24 00 26 00 f6
120 4e ad ff 22 22 02 4e ad ff dc 22 4d 4e ae fe 62 dc
130 43 fa ff bc 20 3c 00 00 01 61 4e ee ff 2e 64 6f b2
140 73 2e 6c 69 62 72 61 72 79 00 4e 49 4c 3a 00 00 bc
150 00 00 03 f2                                     c3
```

Lauffähige Hex-Version von RunBack



Überprüfen mit Checksum (Als Sprache BASIC vermerken)

```
*                                                       042
* RunBack.asm - run executable as independent
  background process                                    442
* Copyright (C) 1988 Ralph Babel, Falkenweg 3,
  D-6204 Taunusstein                                    555
* all rights reserved - alle Rechte vorbehalten         899
*                                                       489
* 31-Mar-1988 created                                   819
*                                                       449
                                                        449
        nolist                                          198
                                                        198
        include "exec/types.i"                          138
        include "exec/memory.i"                         018
        include "exec/ports.i"                          761
        include "exec/strings.i"                        630
        include "libraries/dos.i"                       771
        include "libraries/dosextens.i"                 331
                                                        331
        list                                            755
                                                        755
*                                                       665
* registers                                             527
*                                                       749
                                                        749
REG_SysBase     equr    a6                              540
REG_DOSBase     equr    a5                              477
REG_Process     equr    a4                              087
REG_ArgLine     equr    a3                              971
REG_MemArea     equr    a2                              760
                                                        760
REG_Result      equr    d7                              256
REG_ConsoleTask equr    d6                              290
                                                        290
*                                                       764
```

▶

```
* macros                                                    687
*                                                           781
                                                            781
xlib    macro                                               246
        xref    _LVO\1                                      820
        endm                                                372
                                                            372
callsys macro                                               797
        jsr     _LVO\1(REG_SysBase)                         813
        endm                                                101
                                                            101
jumpsys macro                                               170
        jmp     _LVO\1(REG_SysBase)                         446
        endm                                                926
                                                            926
calldos macro                                               166
        jsr     _LVO\1(REG_DOSBase)                         917
        endm                                                845
                                                            845
*                                                           787
* IMPORT, external symbol references                        307
*                                                           985
                                                            985
        xlib    OpenLibrary                                 407
        xlib    CloseLibrary                                909
        xlib    AllocMem                                    070
        xlib    FreeMem                                     186
        xlib    FindTask                                    593
        xlib    WaitPort                                    926
        xlib    GetMsg                                      748
        xlib    ReplyMsg                                    782
        xlib    Forbid                                      663

        xlib    Open                                        588
        xlib    Close                                       225
        xlib    CreateProc                                  528
        xlib    Execute                                     210
                                                            210
*                                                           724
* constants                                                 135
*                                                           797
                                                            797
AbsExecBase     equ     4                                   934
                                                            934
CMD_BUF_SIZ     equ     256     ;won't check for
  limit!!!                                                  001
                                                            001
*                                                           055
* code section                                              997
*                                                           715
                                                            715
        section "text",code                                 448
                                                            448
*                                                           770
* main entry point                                          385
*                                                           951
                                                            951
        movea.l a0,REG_ArgLine                              441
                                                            441
        movea.l AbsExecBase,REG_SysBase                     668
                                                            668
        suba.l  a1,a1                                       693
        callsys FindTask                                    908
                                                            908
        movea.l d0,REG_Process                              610
        tst.l   pr_CLI(REG_Process)                         265
        bne.s   fromCLI                                     299
                                                            299
*                                                           881
* from Workbench, huh?                                      158
*                                                           048
                                                            048
        lea     pr_MsgPort(REG_Process),a0                  613
        callsys WaitPort                                    846
        lea     pr_MsgPort(REG_Process),a0                  083
        callsys GetMsg                                      001
        move.l  d0,d2                                       252
        callsys Forbid  ;probably don't need this
         one because of 'jumpsys'                           543
        movea.l d2,a1                                       541
        jumpsys ReplyMsg                                    567
                                                            567
*                                                           269
* from CLI                                                  359
*                                                           613
                                                            613
fromCLI                                                     961
        moveq   #RETURN_FAIL,REG_Result ;no dos?!?          196
                                                            196
        lea     dosName(pc),a1                              103
        moveq   #LIBRARY_VERSION,d0                         564
        callsys OpenLibrary                                 418
        tst.l   d0                                          075
        beq.s   6$                                          886
        movea.l d0,REG_DOSBase                              958
                                                            958
        moveq   #RETURN_ERROR+2,REG_Result      ;no
         mem!?!                                             863
                                                            863
        move.l  #childEnd-childStart+CMD_BUF_SIZ,d0         357
        moveq   #MEMF_PUBLIC,d1                             732
        callsys AllocMem                                    505
        move.l  d0,d3   ;used for CreateProc()              313
        beq.s   5$                                          535
                                                            535
        movea.l d0,REG_MemArea                              402
                                                            402
        moveq   #RETURN_ERROR+1,REG_Result
         ;cannot "CreateProc()"                              153
                                                            153
        lea     childStart(pc),a0                           526
        movea.l d0,a1                                       806
        moveq   #childEnd-childStart-1,d0                   193
1$      move.b  (a0)+,(a1)+                                 068
        dbf     d0,1$                                       305
                                                            305
        moveq   #LF,d0                                      506
        bra.s   3$                                          091
                                                            091
2$      move.b  (REG_ArgLine)+,(a1)+                        945
3$      cmp.b   (REG_ArgLine),d0                            964
        bne.s   2$                                          488
                                                            488
        clr.b   (a1)                                        764
                                                            764
        move.l
          pr_ConsoleTask(REG_Process),REG_ConsoleTas
        k                                                   116
        clr.l   pr_ConsoleTask(REG_Process)                 719
                                                            719
* now prepare parameters for CreateProc()                   408
                                                            408
        lea     runBackName(pc),a0                          102
        move.l  a0,d1                                       261
                                                            261
        moveq   #0,d2                                       526
        move.b  LN_PRI(REG_Process),d2                      264
                                                            264
* create fake SegList pointer                               829
                                                            829
        lsr.l   #2,d3                                       232
        subq.l  #1,d3                                       628
                                                            628
        move.l  #4000,d4        ;much too much              919
                                                            919
        calldos CreateProc                                  029
                                                            029
        move.l
          REG_ConsoleTask,pr_ConsoleTask(REG_Process
        )                                                   437
                                                            437
        tst.l   d0                                          702
        beq.s   4$                                          023
                                                            023
        moveq   #RETURN_OK,REG_Result                       351
        bra.s   5$                                          146
                                                            146
4$      movea.l REG_MemArea,a1                              579
        move.l  #childEnd-childStart+CMD_BUF_SIZ,d0         105
        callsys FreeMem                                     449
                                                            449
5$      movea.l REG_DOSBase,a1                              944
        callsys CloseLibrary                                898
                                                            898
6$      move.l  REG_Result,d0                               381
        rts                                                 882
                                                            882
runBackName:                                                731
        dc.b    'Ralph Babel''s "RunBack"',0    ;ego
         boosting                                           233
                                                            233
        cnop    0,2                                         023
                                                            023
*                                                           341
* code for the child process                                469
*                                                           043
                                                            043
childStart:                                                 283
                                                            283
        movea.l AbsExecBase,REG_SysBase                     518
                                                            518
        lea     dosName(pc),a1                              377
        moveq   #LIBRARY_VERSION,d0                         966
        callsys OpenLibrary                                 540
        tst.l   d0                                          005
        beq.s   2$                                          572
                                                            572
        movea.l d0,REG_DOSBase                              596
                                                            596
        lea     nilName(pc),a0                              935
        move.l  a0,d1                                       502
        move.l  #MODE_NEWFILE,d2                            018
        calldos Open                                        294
        tst.l   d0                                          079
        beq.s   1$                                          953
                                                            953
        lea     childEnd(pc),a0                             275
        move.l  a0,d1                                       442
        move.l  d0,d2                                       741
        move.l  d0,d3                                       137
        calldos Execute                                     094
                                                            094
        move.l  d2,d1                                       858
        calldos Close                                       510
                                                            510
1$      movea.l REG_DOSBase,a1                              677
        callsys CloseLibrary                                111
                                                            111
2$      lea     childStart(pc),a1                           015
        move.l  #childEnd-childStart+CMD_BUF_SIZ,d0         941
        jumpsys FreeMem                                     333
                                                            333
dosName:                                                    106
        dc.b    'dos.library',0                             856
                                                            856
nilName:                                                    206
        dc.b    'NIL:',0                                    924
                                                            924
childEnd:                                                   549
                                                            549
*                                                           035
* parent task will copy parameters in this area
  (allocated copy, of course)                               772
*                                                           886
                                                            886
        end                                                 109
```

Assembler-Code für Hintergründiges

```
;                                                           059
;RunBack.bat                                                585
;                                                           520
FailAt 1                                                    586
Assem RunBack.asm -o T:RunBack.obj -i MCC-Assem:INCLUDE     430
BLink T:RunBack.obj TO RunBack LIB LIB:amiga.lib ND         978
FailAt 10                                                   396
```

Batchfile zur Erzeugung von RunBack

Patchen in BASIC Scrollen ohne Leiden

Die Scroll-Funktion des Amiga-Betriebssystems (nicht des Amiga Basics!) hat eine unangenehme Macke: Falls man pur horizontal scrollt (dy = 0), gehen bei jedem Aufruf einige Bytes Speicher unwiederbringlich verloren, typischerweise 128 Stück. Das muß in einer Multitasking-Umgebung natürlich vermieden werden.

Zum Glück ist der Fehler aber erkannt und kann umgangen werden. Abhilfe besteht darin, dem zuständigen Screen eine eigene TmpRas-Struktur zu beschaffen, die er von Haus aus nicht hat, weil sie normalerweise nur beim Flächenfüllen und Ähnlichem gebraucht wird.

Nach mehrtägigem Studium der ROM-Kernal-Manuals weiß man, wie das mit der TmpRas-Struktur in der Praxis gemacht werden muß. Hier sollen keine Einzelheiten breitgetreten werden, nur das Ergebnis. Listing 1 ist ein Programm, das eine TmpRas-Struktur anlegt und das Scrollen ausprobiert. Zur Demonstration wird die TmpRas-Struktur beim ersten Mal (pa = 0) nicht aktiviert (per POKEL), so daß der Speicherverlust in der Anzeige verfolgt werden kann. Erst im zweiten Durchlauf wird die Adresse der TmpRas-Struktur tatsächlich per POKEL in die RastPort-Struktur eingetragen, die Speicheranzeige bleibt danach stabil.

Wenn dieser Patch angewendet werden soll, müssen die verschiedenen Phasen eingehalten werden: 1. Einmal vorweg muß der Speicher reserviert werden; 2. Wenn dann ein bestimmter Screen ausersehen ist, muß die TmpRas-Struktur angelegt werden, deren Adresse schließlich in die RastPort-Struktur des Screens einzutragen ist; 3. Am Schluß muß, *bevor* der Screen (oder das Programm) geschlossen wird, die TmpRas-Struktur beim RastPort abgemeldet und der Speicher wieder freigegeben werden.

Der einzige Punkt, der noch etwas unklar ist, besteht in der nötigen Speichergröße für die TmpRas-Struktur. Beim Flächenflüllen lautet die Regel, daß ausreichend Platz für *eine* Bitplane in der Größe der zu füllenden Fläche benötigt wird; zum Scrollen fehlen Angaben. Die Erfahrung sagte, daß man normalerweise 128 Byte bei jedem Scroll-Aufruf verlor. Zur Sicherheit wurde das Doppelte genommen. Wenn jemand Genaueres herausfindet, sollte er es schnell der Redaktion melden.

(Dr. Peter Kittel)


Auf Leserdiskette



Überprüfen mit Checksum

```
'" Scrolltest                                      039
'" P. Kittel, Ffm 10.2.88, 19.3.88                 546
'" Fehler:  bei rein horizontalem                  993
'"          Scrollen verliert man bei              660
'"          jedem Aufruf unwiederbringlich         427
'"          Speicher.                              494
'" Abhilfe: dem RastPort muß ein TmpRas             317
'"          zur Verfügung gestellt                 064
'"          werden (siehe ROM Kernal               871
'"          Man.).                                 122
                                                   122
rp&=WINDOW(8)                                      668
LOCATE 2,1                                         303
PRINT "Vorheriger
 TmpRas-Pointer:";PEEKL(rp&+12) '" &TmpRas         418
                                                   418
DECLARE FUNCTION AllocMem& LIBRARY                 413
'"                FreeMem                          312
DECLARE FUNCTION InitTmpRas& LIBRARY               328
                                                   328
LIBRARY "exec.library"                             021
LIBRARY "graphics.library"                         594
                                                   594
ras&=AllocMem&(520,3) '" MEMF_PUBLIC |
 MEMF_CHIP                                         193
TmpRas&=InitTmpRas&(ras&+512,ras&,512)             100
                                                   100
FOR pat=0 TO 1 '" Schleife: erst ohne, dann
 mit Patch                                         936
  POKEL rp&+12,TmpRas&*pat '" Patch nur bei
   pat=1 aktiv                                     484
                                                   484
  '" Bild füllen                                   187
  CLS                                              345
  LOCATE 5,1                                       631
  FOR i=1 TO 17                                    380
    FOR j=32 TO 107:PRINT CHR$(j);:NEXT            573
    PRINT                                          942
    NEXT                                           869
                                                   869
  '" Scrollen                                      096
  f=3                                              666
  LOCATE 1,1:PRINT "Scrollweite in
   Bits[2SPACES]:";f;                              967
  PRINT SPC(10);                                   451
  IF pat=0 THEN                                    443
    PRINT "OHNE Patch verliert man
     Speicher."                                    692
          ELSE                                     709
    PRINT "MIT Patch bleibt der Speicher
     unberührt."                                   148
    END IF                                         250
  LOCATE 2,1:PRINT "Freier
   Systemspeicher:";FRE(-1)                        374
  FOR i=1 TO 620/f                                 856
    LOCATE 3,1:PRINT "Freier
     Systemspeicher:";FRE(-1)                      209
    SCROLL (0,28)-(620,180),-f,0                   453
    NEXT                                           852
  LOCATE 3,1:PRINT "Freier
   Systemspeicher:";FRE(-1)                        413
                                                   413
  NEXT pat                                         861
                                                   861
'" Aufräumen                                       992
POKEL rp&+12,0                                     965
FreeMem ras&,520                                   125
LIBRARY CLOSE                                      772
SYSTEM                                             797
                                                   797
```

Speicherkiller in BASIC gezähmt.

Vom Amiga zum PC

Amiga-Anwender arbeiten nicht selten zusätzlich an einem MS-DOS-Rechner und wollen daher Daten zum PC übertragen. „A2PC" hilft dabei.

Sofern Sie nicht glücklicher Besitzer eines Amiga 2000 mit ausnahmsweise funktionsfähiger PC-Steckkarte sind, so mußten Sie bisher andere Mittel zu Hilfe nehmen, um Ihre Daten über den doch recht breiten Graben zu bringen, der Amiga und PC trennt.

Bestens hierfür geeignet ist das Programm „DOS to DOS" von Central Coast Software, das den Austausch von oder auf Atari-Disketten gleichermaßen beherrscht und daher mit gutem Gewissen empfohlen werden kann. Während die Anpassung der unterschiedlichen Zeilentrenner, Linefeed beim Amiga, Carriage Return unter MS-DOS, zwar möglich ist, übernimmt „DOS to DOS" eines jedoch nicht, und das ist mindestens gleichermaßen störend: die Anpassung der beiden differierenden Zeichensätze aneinander.

Während der Amiga den Standard-ASCII-Zeichensatz und auch die entsprechenden ANSI-Steuersequenzen verwendet, übersah IBM in unergründlicher Weisheit, bei Einführung ihres PC, diesen bereits seit 1979 als Normvorschlag vorliegenden internationalen 8-Bit-Schriftsatz, und glaubte ein eigenes Konglomerat von Sonderzeichen für den internationalen Gebrauch entwerfen zu müssen. Bei der notwendigen Umwandlung von Amiga-Texten in etwas, das ein PC verdauen kann, hilft Ihnen „A2PC". Es wandelt deutsche Umlaute und „Eszet" vom Amiga-ASCII in PC-ASCII um und beachtet auch, daß PCs üblicherweise die Sequenz CR/LF anstelle von LF benutzen. An das Ende des Textes wird noch das berüchtigte CTRL-Z angefügt, um auch dem begriffsstutzigsten Programm klarzumachen, daß die Datei hiermit beendet ist.

Um Amiga-Textdateien, die mit Hilfe eines Editors wie etwa „Ed", „TxEd" oder „MicroEmacs" erstellt worden sind, für eine leichtere Weiterverarbeitung mit einem „wirklichen" Textverarbeitungsprogramm vorzubereiten, wird außerdem jedes LF, dem nicht ein weiteres Linefeed, ein Leerzeichen oder das Dateiende folgt, wie ein Leerzeichen behandelt. Dadurch werden nicht-eingerückte Absätze, die durch LFs in einzelne Zeilen getrennt sind, zu einem durchgängigen Absatz verschmolzen, so daß eine Textverarbeitung etwa automatischen Zeilenumbruch durchführen kann. Wer schon einmal einen solcherart vorformatierten Text von Hand bearbeiten mußte, der weiß, welche Arbeit er sich nun ersparen kann. Wird dieses Feature nicht gewünscht, etwa bei Tabellen oder anderen speziellen Formatierungen, so kann durch Angabe der Kommandozeilenoption „NOJOIN" dieses automatische Zusammenfügen unterbunden werden. Eine so behandelte Datei kann nun wie gewohnt mittels „DOS to DOS", per serielle oder parallele Schnittstelle oder auf welchem Wege auch immer zu IBM übertragen werden. Beachten Sie, daß die Umwandlung von CR bereits durch „A2PC" durchgeführt wird und daher in „D2D" beim „COPY" nicht „-A" spezifiziert werden darf. Dies eliminiert dann auch das Problem der Anpassung von Zeichen außerhalb des 7-Bit-Bereichs von 0 bis 127.

Zum Programm selbst: Um einmal etwas Abwechslung in die triste Welt von Assembler, BASIC und C des Amiga-Programmierers zu bringen, wurde dieses Programm in BCPL geschrieben. Da BCPL-Programme auf dem Amiga nicht in vergleichbarer Weise dem Effekt üblicher C-Compiler unterliegen, der auch kleine Programme nicht selten auf bis zu 16 Kilobytes Executable aufbläht, wird zusätzlich das fertige Programm abgedruckt. Es hat eine Länge von nur 1320 Bytes und kann nach Eingabe mit NanoMon wie gewohnt vom CLI aufgerufen werden. Dem Aufruf müssen die Namen des Quellfiles und der zu erzeugenden Datei übergeben werden, der Inhalt einer unter dem Namen des Targetfiles bereits existierenden Datei wird gelöscht! Werden die bei Einhaltung der Parameterreihenfolge optionalen Schlüsselworte „FROM" und „TO" angegeben, so kann auch zuerst die Zieldatei genannt werden. „NOJOIN" ist ein Switch (Schalter), der angegeben werden kann, und dann das automatische Verketten angrenzender Zeilen unterbindet.

(Ralph Babel)

DOS to DOS
Info: DTM, Poststr. 25, 6200 Wiesbaden, Tel.: 0 61 21/56 00 84


Auf Leserdiskette



Eingabe mit Nanomon

Sprache: BCPL
Programmname: A2PC

```
000 00 00 03 f3 00 00 00 00 00 00 00 02 00 00 00 00 1c
010 00 00 00 01 00 00 00 4f 00 00 00 ee 00 00 03 e9 1b
020 00 00 00 4f 28 6a 01 64 70 0c 4e 95 24 01 22 3c 96
030 00 00 00 95 49 fa ff ee 28 6c ff fc 2f 0c 2f 02 63
040 d9 cc d9 cc 20 0c 67 14 20 2c 00 04 e5 80 b2 b4 90
050 08 00 6c 04 22 34 08 00 28 54 60 e4 2c 01 06 81 53
060 00 00 00 32 91 c8 28 6a 00 74 70 0c 4e 95 4a 81 d5
070 67 00 00 e0 06 81 00 00 00 32 2a 2a 00 70 24 41 28
080 d5 ca d5 ca 2e 01 20 4a 20 c6 28 46 d9 c7 d9 cc 50
090 d9 cc 20 3c 47 4c 00 03 20 c0 54 80 b1 cc 6f f8 fc
0a0 91 c8 20 2f 00 0c 43 ef 00 10 93 c0 23 7c ff ff 84
0b0 ff ff 00 04 22 09 04 80 00 00 00 a0 d0 81 23 40 14
0c0 00 08 e4 81 25 41 00 30 2e 1f 2c 07 e5 86 de b0 c0
0d0 68 00 e5 87 58 86 be 86 6d 12 22 30 68 00 28 45 e5
0e0 70 0c 4e 95 4a 81 67 00 00 6c 60 e8 22 1f 28 45 a5
0f0 70 0c 4e 95 4a 81 67 5e 28 6a 02 18 70 0c 4e 95 99
100 e5 81 26 41 2f 0b 67 0c 49 ea 02 18 70 0f 28 db c2
110 51 c8 ff fc 49 fa 00 44 25 4c 00 08 28 6a 00 04 f0
120 70 20 72 00 4e 95 70 00 2e 00 22 1f 67 0e 26 41 af
130 49 ea 02 18 70 0f 26 dc 51 c8 ff fc 22 0a e4 81 7f
140 04 81 00 00 00 32 28 6a 00 78 70 0c 4e 95 20 07 2c
150 4e 75 4a 9f 4a 9f 70 ff 4e 75 20 01 60 ca 00 00 c4
160 00 00 03 f2 00 00 03 e9 00 00 00 ee 00 00 00 ee fc
170 60 00 00 3e 22 3c 00 00 00 8e 4e d6 22 3c 00 00 b8
180 00 99 4e d6 22 3c 00 00 00 9a 4e d6 22 3c 00 00 c0
190 00 84 4e d6 22 3c 00 00 00 94 4e d6 22 3c 00 00 fa
1a0 00 81 4e d6 22 3c 00 00 00 e1 4e d6 22 11 4e d6 da
1b0 22 11 0c 81 00 00 00 c4 67 ba 0c 81 00 00 00 d6 6c
1c0 67 ba 0c 81 00 00 00 dc 67 ba 0c 81 00 00 00 df 84
```

```
1d0 67 d2 0c 81 00 00 00 e4 67 b2 0c 81 00 00 00 f6 90
1e0 67 b2 0c 81 00 00 00 fc 67 b2 60 c0 72 0a 22 81 51
1f0 74 ff 23 42 00 04 76 24 d6 89 e4 8b 23 43 00 20 0e
200 76 40 24 29 00 20 47 ec 02 94 22 0b e4 89 20 3c c8
210 00 00 01 34 28 6a 01 38 4e 95 4a 81 67 00 02 3c d6
220 22 29 00 20 e5 89 22 30 18 00 20 3c 00 00 01 34 30
230 28 6a 00 ec 4e 95 23 41 00 10 4a 81 67 00 01 e0 e5
240 20 3c 00 00 01 34 28 6a 01 04 4e 95 23 41 00 08 71
250 22 29 00 10 20 3c 00 00 01 34 28 6a 00 f4 4e 95 a4
260 22 29 00 20 e5 89 22 30 18 04 20 3c 00 00 01 34 88
270 28 6a 00 f0 4e 95 23 41 00 14 4a 81 67 00 01 48 69
280 20 3c 00 00 01 34 28 6a 01 08 4e 95 23 41 00 0c cd
290 22 29 00 14 20 3c 00 00 01 34 28 6a 00 f8 4e 95 80
2a0 20 3c 00 00 01 34 28 6a 00 d8 4e 95 23 41 00 18 7e
2b0 60 00 00 b4 22 29 00 20 e5 89 4a b0 18 08 66 00 7a
2c0 00 70 20 3c 00 00 01 34 28 6a 00 d8 4e 95 23 41 c0
2d0 00 18 74 00 76 0a b6 81 66 02 46 82 84 a9 00 04 37
2e0 78 00 7a 20 ba 81 66 02 46 84 84 84 78 00 7c ff 6a
2f0 bc 81 66 02 46 84 84 84 23 42 00 04 4a 82 67 00 58
300 00 22 72 0d 20 3c 00 00 01 34 28 6a 00 e0 4e 95 56
310 72 0a 20 3c 00 00 01 34 28 6a 00 e0 4e 95 60 00 9e
320 00 10 72 20 20 3c 00 00 01 34 28 6a 00 e0 4e 95 f1
330 72 0a b2 a9 00 18 66 00 00 16 72 0d 20 3c 00 00 68
340 01 34 28 6a 00 e0 4e 95 72 0a 60 00 00 28 42 a9 d4
350 00 04 22 29 00 18 20 3c 00 00 01 34 49 ec ff 84 18
360 4e 95 60 00 00 10 22 29 00 18 74 0a b2 82 67 00 ac
370 ff 44 60 da 23 41 00 1c 74 ff b4 81 67 00 00 0e 54
380 20 3c 00 00 01 34 28 6a 00 e0 4e 95 72 ff b2 a9 49
390 00 1c 66 00 ff 0c 72 1a 20 3c 00 00 01 34 28 6a ec
3a0 00 e0 4e 95 42 91 20 3c 00 00 01 34 28 6a 01 00 c7
3b0 4e 95 22 29 00 0c 20 3c 00 00 01 34 28 6a 00 f8 c3
3c0 4e 95 60 00 00 3a 22 29 00 20 e5 89 24 30 18 04 d1
3d0 47 ec 02 ac 22 0b e4 89 20 3c 00 00 01 34 28 6a fa
3e0 01 28 4e 95 72 00 20 3c 00 00 01 40 28 6a 00 28 46
3f0 4e 95 20 3c 00 00 01 34 28 6a 01 a0 4e 95 20 3c ec
400 00 00 01 34 28 6a 00 fc 4e 95 22 29 00 08 20 3c fd
410 00 00 01 34 28 6a 00 f4 4e 95 60 00 00 3a 22 29 a8
420 00 20 e5 89 24 30 18 00 47 ec 02 cc 22 0b e4 89 23
430 20 3c 00 00 01 34 28 6a 01 28 4e 95 72 00 20 3c 70
440 00 00 01 40 28 6a 00 28 4e 95 20 3c 00 00 01 34 8f
450 28 6a 01 a0 4e 95 60 00 00 16 47 ec 02 ec 22 0b 0f
460 e4 89 20 3c 00 00 01 34 28 6a 01 24 4e 95 22 11 d5
470 20 3c 00 00 01 34 28 6a 00 08 4e 95 4e d6 4e 71 3c
480 14 46 52 4f 4d 2f 41 2c 54 4f 2f 41 2c 4e 4f 4a 26
490 4f 49 4e 2f 53 00 00 00 1d 43 61 6e 27 74 20 6f 16
4a0 70 65 6e 20 22 25 53 22 20 66 6f 72 20 6f 75 74 40
4b0 70 75 74 20 2d 20 00 00 1c 43 61 6e 27 74 20 6f cd
4c0 70 65 6e 20 22 25 53 22 20 66 6f 72 20 69 6e 70 a5
4d0 75 74 20 2d 20 00 00 00 3b 55 73 61 67 65 3a 20 b5
4e0 41 32 50 43 20 5b 46 52 4f 4d 5d 20 3c 61 6d 69 97
4f0 67 61 2d 61 73 63 69 69 3e 20 5b 54 4f 5d 20 3c 9a
500 69 62 6d 2d 61 73 63 69 69 3e 20 5b 4e 4f 4a 4f 66
510 49 4e 5d 0a 00 00 00 00 00 00 00 01 00 00 00 80 69
520 00 00 00 68 00 00 03 f2                         78
```

Superkurz: lauffähiges BCPL-Compilat

```
//
// A2PC.b - Convert AMIGA-ASCII (ANSI X3.64-1979) to
   IBM-PC-ASCII
// Copyright (C) 07-Mar-1988 by Ralph Babel,
   Falkenweg 3, D-6204 Taunusstein
// all rights reserved - alle Rechte vorbehalten
//

GET "LIBHDR"
GET "CLIHDR"

MANIFEST
$(
   AMY.UC.A.UMLAUT = #XC4
   AMY.UC.O.UMLAUT = #XD6
   AMY.UC.U.UMLAUT = #XDC
   AMY.LC.A.UMLAUT = #XE4
   AMY.LC.O.UMLAUT = #XF6
   AMY.LC.U.UMLAUT = #XFC
   AMY.ESZET       = #XDF

   IBM.UC.A.UMLAUT = 142
   IBM.UC.O.UMLAUT = 153
   IBM.UC.U.UMLAUT = 154
   IBM.LC.A.UMLAUT = 132
   IBM.LC.O.UMLAUT = 148
   IBM.LC.U.UMLAUT = 129
   IBM.ESZET       = 225
   IBM.EOF         = 26

   ASC.LF = 10
   ASC.CR = 13

   arg.upb = 64 // argument upper bound

   argv.from   = 0
   argv.to     = 1
   argv.nojoin = 2
$)

// code section

LET convert(ch) = VALOF
$(
   SWITCHON ch INTO
   $(
      CASE AMY.UC.A.UMLAUT:
       RESULTIS IBM.UC.A.UMLAUT
      CASE AMY.UC.O.UMLAUT:
       RESULTIS IBM.UC.O.UMLAUT
      CASE AMY.UC.U.UMLAUT:
       RESULTIS IBM.UC.U.UMLAUT
      CASE AMY.LC.A.UMLAUT:
       RESULTIS IBM.LC.A.UMLAUT
      CASE AMY.LC.O.UMLAUT:
       RESULTIS IBM.LC.O.UMLAUT
      CASE AMY.LC.U.UMLAUT:
       RESULTIS IBM.LC.U.UMLAUT
      CASE AMY.ESZET:
       RESULTIS IBM.ESZET
      DEFAULT:
       RESULTIS ch
   $)
$)

// main entry point

LET start() BE
$(
   LET rc = return.hard
   LET flag = TRUE
   LET termin, termout = ?, ?
   LET filein, fileout = ?, ?
   LET rch, wch = ?, ?
   LET argv = VEC arg.upb

   TEST rdargs("FROM/A,TO/A,NOJOIN/S", argv,
    arg.upb) THEN
   $(
      filein := findinput(argv!argv.from)
      TEST filein THEN
      $(
         termin := input()
         selectinput(filein)

         fileout := findoutput(argv!argv.to)
         TEST fileout THEN
         $(
            termout := output()
            selectoutput(fileout)

            $(
               wch := VALOF
               $(
                  rch := rdch()
                  SWITCHON rch INTO
                  $(
                     CASE '*N':
                        IF ~argv!argv.nojoin THEN
                        $(
                           rch := rdch()
                           flag := flag \/ rch =
                            '*N' \/ rch = '*S' \/
                            rch = ENDSTREAMCH
                           TEST flag THEN
                           $(
                              wrch(ASC.CR)
                              wrch(ASC.LF)
                           $)
                           ELSE
                           $(
                              wrch('*S')
                           $)
                        $)

                        IF rch = '*N' THEN
                        $(
                           wrch(ASC.CR)
                           RESULTIS ASC.LF
                        $)
                     DEFAULT:
                        flag := FALSE
                        RESULTIS convert(rch)
                  $)
               $)

               IF wch ~= ENDSTREAMCH THEN
               $(
                  wrch(wch)
               $)

            $) REPEATUNTIL wch = ENDSTREAMCH

            wrch(IBM.EOF)

            rc := return.ok

            endwrite()
            selectoutput(termout)
         $)
         ELSE
         $(
            writef("Can't open *"%S*" for output -
             ", argv!argv.to)
            fault(result2(FALSE))
         $)

         endread()
         selectinput(termin)
      $)
      ELSE
      $(
         writef("Can't open *"%S*" for input - ",
          argv!argv.from)
         fault(result2(FALSE))
      $)
   $)
   ELSE
   $(
      writes("Usage: A2PC [FROM] <amiga-ascii> [TO]
       <ibm-ascii> [NOJOIN]*N")
   $)

   stop(rc)
$)
```

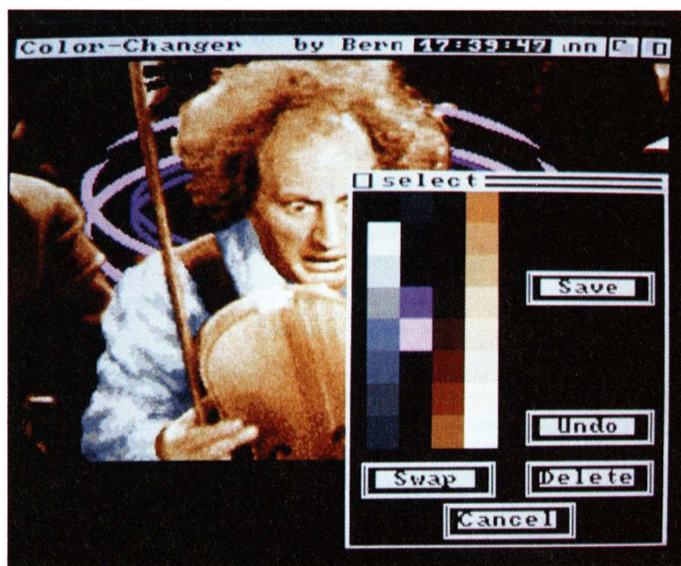
So übersichtlich sind BCPL-Listings

Farben frisch sortiert



Kein Programm ist perfekt. Auch DPaint weist kleine Schwächen auf. Eine davon beheben wir mit unserem Programm. ColorChanger erweitert die Möglichkeiten, die Farbpalette zu manipulieren.

Nach stundenlanger Arbeit an einem Computergemälde kann es sein, daß man mit der Anordnung der Farbpalette nicht zufrieden ist. Sei es, daß Farben, die zum Colorcycling benutzt werden sollen, nicht nebeneinander liegen oder daß bestimmte Farben ineinander verlaufen sollen. DPaint-Features wie Dither und Range lauten die Stichworte. Zwar erlaubt Dpaint in der Palette den Austausch von Farben, jedoch verändert sich damit auch das Aussehen des Bildes. Grüne Dächer und violette Bäume können die Folge sein. Zum Aufräumen von verwilderten Colorpaletten bietet AmigaWelt deshalb den ColorChanger an. Mit diesem Programm ist es möglich, die Reihenfolge der Farben in der Farbpalette eines IFF-Bildes nach Belieben zu vertauschen, ohne das Aussehen des Bildes zu verändern. Das Bitmuster des Bildes wird automatisch an die neue Farbpalette angepaßt. Weiterhin lassen sich Farben durch andere, zum Beispiel ähnliche ersetzen, wodurch Platz für zusätzliche Nuancen geschaffen wird, ohne daß das Erscheinungsbild der Grafik wesentlich verändert wird.



So übersichtlich präsentiert sich ColorChanger dem Anwender

Nach Starten des Programmes aus dem CLI muß über die Tastatur der Name des IFF-Bildes eingegeben werden, die weitere Bedienung erfolgt durch die Maus. Dem Benutzer stehen die beiden Funktionselemente SWAP und DELETE zur Verfügung.

SWAP
vertauscht die Position zweier Farben, welche nacheinander angeklickt werden, in der Palette.

DELETE
verlangt zuerst die Angabe der Farbe, die gelöscht werden soll. Dann muß die Farbe, welche statt der zu löschenden im Bild dargestellt werden soll, ausgewählt werden. Alle Punkte im Bild, welche vorher die zu löschende Farbe hatten, haben nach der Ausführung von DELETE die Farbe, die als zweite angeklickt wurde.

Beide Funktionen können vor der eigentlichen Ausführung noch mit CANCEL abgebrochen werden. Um ein versehentliches Löschen einer Farbe wieder rückgängig zu machen, kann UNDO angeklickt werden. UNDO tauscht den Inhalt des Bildspeichers mit dem des Undo-Speichers aus. Durch wiederholtes Anklicken von UNDO kann man also im schnellen Vergleich eine Änderung direkt auf ihre Wirkung hin testen.

Das Ergebnis seiner Arbeit kann der Grafiker dann mittels SAVE als IFF-Bild unter dem Namen, unter dem es geladen wurde, abspeichern. Die ursprüngliche Version ist damit unwiederbringlich verloren. Das geänderte Bild läßt sich nun mit den gängigen Malprogrammen weiter bearbeiten und der Kreativität des Farbkünstlers steht nichts mehr im Wege. Aus Platzgründen wurde auf den Abdruck des Assemblersources verzichtet. Interessierte finden den 47K-Quelltext auf der Leserdiskette.

(Bernd Lintermann)


Source und lauffähige Version auf Leserdiskette



Fehlerfrei abtippen mit Nanomon

Programmname: ColorChanger
Sprache: Assembler

```
0000 00 00 03 f3 00 00 00 00 00 00 00 02 00 00 00 00 1c
0010 00 00 00 01 00 00 05 c2 00 00 00 01 00 00 03 e9 12
0020 00 00 05 c2 2f 3c f0 f0 f0 f0 4e b9 00 00 03 28 52
0030 24 3c 00 00 0a be 61 00 0a c2 4e b9 00 00 0c 06 59
0040 22 3c 00 00 0c 3c 61 00 05 24 30 39 00 00 0c 5c cd
0050 0c 40 00 01 66 00 00 08 42 b9 00 00 0d a2 0c 40 3e
0060 00 02 66 00 00 08 42 b9 00 00 0e 22 0c 40 00 03 00
0070 66 00 00 08 42 b9 00 00 0f 22 0c 40 00 04 66 00 e7
0080 00 08 42 b9 00 00 11 22 61 00 02 8c 61 00 07 08 d5
0090 61 00 02 18 61 00 0a b0 61 00 0a 8a 61 00 0b 50 a8
00a0 20 01 02 41 ff b8 66 00 00 54 22 00 02 81 00 00 98
00b0 00 40 67 00 ff e8 20 43 30 28 00 26 67 00 ff de 62
00c0 0c 40 00 22 67 00 00 66 0c 40 00 23 67 00 01 0c ae
00d0 0c 40 00 21 67 00 01 7c 0c 40 00 24 67 00 00 d4 29
00e0 60 00 ff ba 43 f9 00 00 0c fc 61 00 02 14 20 79 7f
00f0 00 00 04 28 4e ae fe c8 60 00 ff a2 61 00 02 30 45
0100 61 00 09 8a 2c 79 00 00 00 04 22 79 00 00 03 a2 9a
0110 4e ae fe 62 22 79 00 00 03 a6 4e ae fe 62 22 79 b1
0120 00 00 03 9e 4e ae fe 62 60 00 09 ec 43 f9 00 00 e5
0130 16 24 61 00 01 cc 61 00 01 a4 67 00 ff a8 04 40 a8
0140 00 01 33 c0 00 00 03 d2 33 c0 00 00 03 d8 43 f9 dc
0150 00 00 0d 03 61 00 01 aa 61 00 01 82 67 00 ff 86 d2
0160 04 40 00 01 33 c0 00 00 03 d4 33 c0 00 00 03 d6 e2
0170 61 00 0a 6a 43 f9 00 00 0d 18 61 00 01 84 61 00 11
0180 07 52 30 39 00 00 03 d2 e3 40 32 39 00 00 03 d4 75
0190 e3 41 41 f9 00 00 04 b0 34 30 00 00 31 b0 10 00 09
01a0 00 00 31 82 10 00 61 00 05 ee 61 00 0a 18 60 00 45
01b0 ff 34 43 f9 00 00 16 b4 61 00 01 46 61 00 0a 1e 96
01c0 43 f9 00 00 0d 18 61 00 01 38 61 00 00 a4 61 00 d6
01d0 05 c6 61 00 09 f0 60 00 ff 0c 43 f9 00 00 16 6c 9c
01e0 61 00 01 1e 61 00 00 f6 67 00 fe fa 04 40 00 01 52
01f0 33 c0 00 00 03 d2 43 f9 00 00 0d 0d 61 00 01 02 66
0200 61 00 00 da 67 00 fe de 04 40 00 01 33 c0 00 00 7c
0210 03 d4 33 c0 00 00 03 d8 33 c0 00 00 03 d6 61 00 69
0220 09 bc 43 f9 00 00 0d 18 61 00 00 d6 61 00 00 7c 62
0230 61 00 06 a0 32 39 00 00 03 d2 e3 41 41 f9 00 00 86
0240 04 b0 31 90 10 00 61 00 05 4e 61 00 09 78 60 00 ed
0250 fe 94 43 f9 00 00 15 dc 61 00 00 a6 61 00 05 62 fd
0260 61 00 06 24 61 00 07 70 61 00 08 10 60 00 fe 76 10
0270 2e 39 00 00 03 b2 20 79 00 00 07 fe 22 79 00 00 e2
0280 08 0a 10 10 10 d1 12 c0 53 87 66 00 ff f6 3e 3c be
0290 00 1f 41 f9 00 00 04 b0 43 f9 00 00 04 f0 30 10 6c
02a0 30 d1 32 c0 51 cf ff f8 4e 75 2e 39 00 00 03 b2 19
02b0 20 79 00 00 07 fe 22 79 00 00 08 0a 12 d8 53 87 78
02c0 66 00 ff fa 3e 3c 00 1f 41 f9 00 00 04 b0 43 f9 14
02d0 00 00 04 f0 32 d8 51 cf ff fc 4e 75 61 00 09 10 b9
02e0 02 81 00 00 00 40 67 00 ff f4 20 43 30 28 00 26 c6
02f0 67 00 00 0c 0c 40 00 21 64 00 ff e2 4a 40 4e 75 07
0300 2c 79 00 00 03 a2 20 79 00 00 04 28 24 7c ff ff 80
0310 ff ff 4e ee fe ec 2c 79 00 00 03 a2 41 f9 00 00 94
0320 0c 54 4e ae ff 3a 23 c0 00 00 04 24 4e 75 2c 79 a5
0330 00 00 03 a2 20 79 00 00 04 28 4e ae ff b8 61 00 0c
0340 08 9c 20 79 00 00 04 24 4e ee ff be 42 80 2c 79 48
0350 00 00 00 04 43 f9 00 00 03 8c 4e ae fe 68 23 c0 64
0360 00 00 03 a6 2c 79 00 00 00 04 43 f9 00 00 03 7a 18
0370 4e ae fe 68 23 c0 00 00 03 a2 2c 79 00 00 00 04 71
0380 43 f9 00 00 03 6e 4e ae fe 68 23 c0 00 00 03 9e 2e
0390 4e 75 64 6f 73 2e 6c 69 62 72 61 72 79 00 69 6e 33
03a0 74 75 69 74 69 6f 6e 2e 6c 69 62 72 61 72 79 00 e4
03b0 67 72 61 70 68 69 63 73 2e 6c 69 62 72 61 72 79 4e
03c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 87
03d0 00 00 00 00 00 00 00 00 00 01 00 00 00 00 00 00 9d
03e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 41
03f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 63 61 06
0400 6e 6e 6f 74 20 4f 70 65 6e 20 4f 75 74 70 75 74 e2
0410 2d 66 69 6c 65 0a 00 43 6f 6c 6f 72 2d 43 68 61 40
0420 6e 67 65 72 20 20 20 62 79 20 42 65 72 6e 64 20 f0
0430 4c 69 6e 74 65 72 6d 61 6e 6e 00 0a 0a 00 00 00 57
0440 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 58
0450 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 b5
0460 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 12
0470 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 6f
0480 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 cc
0490 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 29
04a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 86
04b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 e3
04c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 40
04d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 9d
04e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 fa
04f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 57
0500 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 9d
0510 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 fa
0520 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 57
0530 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 b4
0540 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 11
0550 00 00 00 00 2c 79 00 00 03 9e 24 3c 00 00 03 ed 0c
0560 4e ae ff e2 23 c0 00 00 03 aa 4e 75 48 e7 7f fe 8f
0570 61 00 ff e2 67 00 05 4e 2c 79 00 00 03 9e 26 3c 18
0580 00 00 00 08 22 39 00 00 03 aa 24 3c 00 00 04 1a 3f
0590 4e ae ff d6 0c b9 46 4f 52 4d 00 00 04 1a 66 00 c4
05a0 05 24 20 39 00 00 04 1e 23 c0 00 00 08 06 61 00 91
05b0 02 60 23 c0 00 00 08 02 26 39 00 00 04 1e 22 39 4b
05c0 00 00 03 aa 24 39 00 00 08 02 2c 79 00 00 03 9e 93
05d0 4e ae ff d6 61 00 02 28 20 79 00 00 08 02 28 48 45
05e0 42 40 10 28 00 14 0c 40 00 06 6d 00 00 06 30 3c 56
05f0 00 06 33 c0 00 00 0c 5c 13 e8 00 16 00 00 03 c4 cd
0600 42 28 00 16 33 e8 00 0e 00 00 03 c2 30 28 00 0c a0
0610 33 c0 00 00 03 c0 32 00 02 41 00 0f 67 00 00 0a da
0620 02 40 ff f0 06 40 00 10 e6 48 33 c0 00 00 03 be cd
0630 20 28 00 08 41 f0 08 0c 23 c8 00 00 03 ca 20 28 8e
0640 00 04 41 f0 08 08 0c 90 42 4f 44 59 66 00 ff f0 76
0650 23 c8 00 00 03 ce 30 39 00 00 03 be 34 39 00 00 1e
0660 03 c2 c0 c2 23 c0 00 00 03 ba c0 f9 00 00 0c 5c 40
0670 23 c0 00 00 03 b2 61 00 01 98 23 c0 00 00 07 fe 4b
0680 20 39 00 00 03 b2 61 00 01 88 23 c0 00 00 08 0a b3
0690 61 00 00 a4 20 79 00 00 03 ce d1 fc 00 00 00 08 43
06a0 3e 39 00 00 03 c2 53 47 36 39 00 00 0c 5c 53 43 5f
06b0 45 f9 00 00 08 0e 2a 52 61 2e 24 cd 51 cb ff f8 20
06c0 08 2c 00 00 00 15 67 00 00 0c 2a 7c 00 00 04 30 cc
06d0 61 00 00 16 51 cf ff d2 08 ac 00 00 00 15 61 00 19
06e0 00 72 4c df 7f fe 4e 75 4a 39 00 00 03 c4 66 00 91
06f0 00 12 30 39 00 00 03 be 53 40 1a d8 51 c8 ff fc 4e
0700 4e 75 34 39 00 00 03 be 4a 42 66 00 00 04 4e 75 15
0710 10 18 6b 00 00 10 48 80 53 42 1a d8 51 c8 ff fa e0
0720 60 00 ff e6 44 00 48 80 12 18 1a c1 53 42 51 c8 e9
0730 ff fa 60 00 ff d4 20 39 00 00 07 fe 41 f9 00 00 c2
0740 08 0e 74 04 20 c0 d0 b9 00 00 03 ba 51 ca ff f6 93
0750 4e 75 20 79 00 00 03 ca d1 fc 00 00 00 08 43 f9 26
0760 00 00 04 b0 34 39 00 00 0c 5c 3e 3c 00 01 e5 6f d9
0770 53 47 42 40 10 18 02 00 00 f0 e9 48 10 18 02 40 c3
0780 0f f0 42 41 12 18 e8 09 02 01 00 0f 80 01 32 c0 d5
0790 51 cf ff e0 4e 75 20 79 00 00 04 24 d1 fc 00 00 58
07a0 00 2c 43 f9 00 00 04 b0 34 39 00 00 0c 5c 30 3c da
07b0 00 01 e5 68 2c 79 00 00 03 a6 4e ae ff 40 4e 75 5c
07c0 22 79 00 00 03 ca d3 fc 00 00 00 08 41 f9 00 00 b0
07d0 04 b0 34 39 00 00 0c 5c 3e 3c 00 01 e5 6f 53 47 42
07e0 30 18 12 00 e8 48 02 00 00 f0 12 c0 10 01 02 00 ee
07f0 00 f0 12 c0 e9 49 12 c1 51 cf ff e6 4e 75 2c 79 e2
0800 00 00 03 9e 22 39 00 00 03 aa 4e ae ff dc 4e 75 c7
0810 2c 79 00 00 00 04 22 3c 00 01 00 02 4e ae ff 3a b8
0820 4e 75 00 00 00 00 00 00 00 00 00 00 00 00 00 00 35
0830 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 df
0840 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 3c
0850 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 99
0860 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 f6
0870 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 53
0880 00 00 00 00 00 00 22 3c 00 00 0c 3c 2c 79 00 00 3b
0890 03 9e 4e ae ff b8 22 3c 00 00 0c 3c 24 3c 00 00 4f
08a0 03 ee 4e ae ff e2 4a 80 67 00 00 0a 23 c0 00 00 c6
08b0 03 ae 4e 75 20 3c 00 00 15 36 61 00 00 d8 0c 9f 6a
08c0 f0 f0 f0 f0 66 00 ff f8 2f 3c f0 f0 f0 f0 60 00 d4
08d0 f8 14 61 00 fe 62 2e 39 00 00 03 ba 53 47 42 46 60
08e0 7a 01 38 39 00 00 0c 5c 42 80 30 04 e5 40 49 f9 47
08f0 00 00 08 0e d9 c0 53 44 42 41 26 64 10 13 c0 05 6e
0900 67 00 00 06 00 41 80 00 e3 59 51 cc ff ee 30 39 3e
0910 00 00 03 d2 b0 01 66 00 00 10 32 39 00 00 03 d6 2e
0920 61 00 00 52 60 00 00 18 30 39 00 00 03 d4 b0 01 8f
0930 66 00 00 0c 32 39 00 00 03 d8 61 00 00 38 e3 4d 4d
0940 52 46 0c 46 00 08 66 00 ff 9a 41 f9 00 00 08 0e 5e
0950 06 98 00 00 00 01 06 98 00 00 00 01 06 98 00 00 1e
0960 00 01 06 98 00 00 00 01 06 90 00 00 00 01 51 cf b2
0970 ff 6e 4e 75 38 39 00 00 0c 5c 53 44 49 f9 00 00 5a
0980 08 0e 26 5c 0d 93 e2 09 64 00 00 04 0d d3 51 cc 15
0990 ff f2 4e 75 23 c0 00 00 15 2e 20 79 00 00 04 28 e9
09a0 22 7c 00 00 15 22 24 7c 00 00 00 00 26 7c 00 00 3e
09b0 15 80 20 3c 00 00 00 00 22 3c 00 00 00 00 24 3c 5b
09c0 00 00 01 40 26 3c 00 00 00 3e 2c 79 00 00 03 a2 30
09d0 4e ae fe a4 4e 75 2c 79 00 00 03 9e 26 39 00 00 54
09e0 03 ce 06 83 00 00 00 08 96 b9 00 00 08 02 2e 03 ff
09f0 d6 b9 00 00 03 b2 23 c3 00 00 04 1e 26 3c 00 00 4a
0a00 00 08 22 39 00 00 03 ae 24 3c 00 00 04 1a 4e ae 2c
0a10 ff d0 20 79 00 00 03 ce 21 79 00 00 03 b2 00 04 7b
0a20 26 07 22 39 00 00 03 ae 24 39 00 00 08 02 4e ae 1c
0a30 ff d0 61 00 fd 02 3e 39 00 00 03 c2 53 47 4b f9 9a
0a40 00 00 08 0e 42 46 3a 39 00 00 0c 5c 53 45 42 83 a0
0a50 36 39 00 00 03 be 24 35 60 00 20 42 d1 c3 2b 88 59
0a60 60 00 22 39 00 00 03 ae 4e ae ff d0 06 46 00 04 6f
0a70 51 cd ff dc 51 cf ff ce 4e 75 2c 79 00 00 03 9e 08
0a80 22 39 00 00 03 ae 4e ae ff dc 4e 75 2c 79 00 00 2d
0a90 00 04 22 79 00 00 07 fe 20 39 00 00 03 b2 4e ae 55
0aa0 ff 2e 22 79 00 00 08 0a 20 39 00 00 03 b2 4e ae 80
0ab0 ff 2e 22 79 00 00 08 02 20 39 00 00 08 06 4e ae 32
0ac0 ff 2e 4e 75 24 3c 00 00 0a ae 61 00 00 2e 60 00 45
0ad0 00 46 57 72 6f 6e 67 20 46 69 6c 65 20 21 21 21 51
0ae0 0a 00 4e 61 6d 65 20 6f 66 20 74 68 65 20 70 69 e2
0af0 63 74 75 72 65 20 3a 20 00 00 2c 79 00 00 03 9e 08
0b00 4e ae ff c4 22 00 20 42 42 83 52 83 4a 18 66 00 74
0b10 ff fa 4e ee ff d0 20 3c f0 f0 f0 f0 b0 9f 66 00 25
```

▶



```
0b20 ff f6 4e 75 23 f9 00 00 04 24 00 00 0c 92 2c 79 94
0b30 00 00 03 a2 41 f9 00 00 0c 74 4e ae ff 34 23 c0 b3
0b40 00 00 04 28 4e 75 23 f9 00 00 04 24 00 00 0c c2 fc
0b50 30 39 00 00 03 c0 32 39 00 00 03 c2 0c 40 01 40 09
0b60 65 00 00 06 30 3c 01 40 0c 41 00 f3 65 00 00 06 60
0b70 32 3c 00 f3 33 c0 00 00 0c a8 33 c1 00 00 0c aa e8
0b80 41 f9 00 00 0c d4 30 39 00 00 0c 5c 48 c0 32 39 cd
0b90 00 00 03 c0 48 c1 34 39 00 00 03 c2 48 c2 2c 79 85
0ba0 00 00 03 a6 4e ae fe 7a 61 00 fb 8c 41 f9 00 00 6c
0bb0 08 0e 43 f9 00 00 0c dc 22 d8 22 d8 22 d8 22 d8 f8
0bc0 22 d8 22 d8 2c 79 00 00 03 a2 41 f9 00 00 0c a4 47
0bd0 4e ae ff 34 23 c0 00 00 04 2c 4e 75 2c 79 00 00 1e
0be0 03 a2 20 79 00 00 04 2c 4e ae ff b8 4e 75 20 79 2e
0bf0 00 00 04 28 20 68 00 56 2c 79 00 00 00 04 4e ae 75
0c00 fe 8c 22 00 4a 80 67 00 00 20 22 40 28 29 00 14 7c
0c10 3a 29 00 18 2c 29 00 1c 2c 79 00 00 00 04 4e ae 82
0c20 fe 86 22 04 34 05 26 06 4e 75 2c 79 00 00 03 9e 12
0c30 4e ae ff ca 22 00 24 3c 00 00 0c 3c 26 3c 00 00 a0
0c40 00 14 4e ae ff d6 41 f9 00 00 0c 3c 10 3c 00 0a 9d
0c50 4a 10 67 00 00 0a b0 18 66 00 ff f6 42 20 4e 75 58
0c60 ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff 2a
0c70 ff ff ff ff 00 ff ff ff 00 00 00 00 01 40 01 00 f6
0c80 00 05 00 01 00 00 00 0f 00 00 00 00 00 00 03 f3 8b
0c90 00 00 00 00 00 00 00 00 00 96 00 0d 00 a0 00 ba 0e
0ca0 00 01 00 00 02 40 00 00 10 0a 00 00 15 54 00 00 08
0cb0 00 00 00 00 0c fc 00 00 00 00 00 00 00 00 00 0a 0d
0cc0 00 0a 01 40 01 00 00 0f 00 00 00 0d 00 a0 00 64 44
0cd0 00 01 00 00 00 00 00 00 08 80 00 00 00 00 00 00 e6
0ce0 00 00 00 00 00 00 00 00 00 00 00 00 0c d4 00 00 b2
0cf0 00 00 ff ff ff ff 00 0f 00 00 00 00 00 00 00 00 62
0d00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 65
0d10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 c2
0d20 73 65 6c 65 63 74 00 73 77 61 70 20 77 69 74 68 ae
0d30 00 72 65 70 6c 61 63 65 20 62 79 00 20 57 4f 52 9b
0d40 4b 49 4e 47 20 00 00 00 0d 62 00 0a 00 0a 00 10 45
0d50 00 10 00 04 00 01 00 01 00 00 0d 4e 00 00 00 00 5b
0d60 00 00 00 00 00 00 00 00 00 00 00 00 00 01 00 00 3c
0d70 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 90
0d80 00 00 00 00 00 00 00 00 0d a2 00 0a 00 1a 00 10 64
0d90 00 10 00 04 00 01 00 01 00 00 0d 8e 00 00 00 00 0f
0da0 00 00 00 00 00 00 00 00 00 00 00 00 00 02 00 00 59
0db0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 04
0dc0 00 01 00 00 00 00 00 00 0d e2 00 0a 00 2a 00 10 81
0dd0 00 10 00 04 00 01 00 01 00 00 0d ce 00 00 00 00 c3
0de0 00 00 00 00 00 00 00 00 00 00 00 00 00 03 00 00 76
0df0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 78
0e00 00 02 00 00 00 00 00 00 0e 22 00 0a 00 3a 00 10 ec
0e10 00 10 00 04 00 01 00 01 00 00 0e 0e 00 00 00 00 bd
0e20 00 00 00 00 00 00 00 00 00 00 00 00 00 04 00 00 7c
0e30 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 d5
0e40 00 03 00 00 00 00 00 00 0e 62 00 0a 00 4a 00 10 09
0e50 00 10 00 04 00 01 00 01 00 00 0e 4e 00 00 00 00 71
0e60 00 00 00 00 00 00 00 00 00 00 00 00 00 05 00 00 99
0e70 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 49
0e80 00 04 00 00 00 00 00 00 0e a2 00 0a 00 5a 00 10 26
0e90 00 10 00 04 00 01 00 01 00 00 0e 8e 00 00 00 00 25
0ea0 00 00 00 00 00 00 00 00 00 00 00 00 00 06 00 00 b6
0eb0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 bd
0ec0 00 05 00 00 00 00 00 00 0e e2 00 0a 00 6a 00 10 43
0ed0 00 10 00 04 00 01 00 01 00 00 0e ce 00 00 00 00 d9
0ee0 00 00 00 00 00 00 00 00 00 00 00 00 00 07 00 00 d3
0ef0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 31
0f00 00 06 00 00 00 00 00 00 0f 22 00 0a 00 7a 00 10 ae
0f10 00 10 00 04 00 01 00 01 00 00 0f 0e 00 00 00 00 d3
0f20 00 00 00 00 00 00 00 00 00 00 00 00 00 08 00 00 d9
0f30 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 8e
0f40 00 07 00 00 00 00 00 00 0f 62 00 1a 00 0a 00 10 5b
0f50 00 10 00 04 00 01 00 01 00 00 0f 4e 00 00 00 00 87
0f60 00 00 00 00 00 00 00 00 00 00 00 00 00 09 00 00 f6
0f70 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 02
0f80 00 08 00 00 00 00 00 00 0f a2 00 1a 00 1a 00 10 78
0f90 00 10 00 04 00 01 00 01 00 00 0f 8e 00 00 00 00 3b
0fa0 00 00 00 00 00 00 00 00 00 00 00 00 00 0a 00 00 13
0fb0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 76
0fc0 00 09 00 00 00 00 00 00 0f e2 00 1a 00 2a 00 10 95
0fd0 00 10 00 04 00 01 00 01 00 00 0f ce 00 00 00 00 ef
0fe0 00 00 00 00 00 00 00 00 00 00 00 00 00 0b 00 00 30
0ff0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 ea
1000 00 0a 00 00 00 00 00 00 10 22 00 1a 00 3a 00 10 00
1010 00 10 00 04 00 01 00 01 00 00 10 0e 00 00 00 00 e9
1020 00 00 00 00 00 00 00 00 00 00 00 00 00 0c 00 00 36
1030 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 47
1040 00 0b 00 00 00 00 00 00 10 62 00 1a 00 4a 00 10 1d
1050 00 10 00 04 00 01 00 01 00 00 10 4e 00 00 00 00 9d
1060 00 00 00 00 00 00 00 00 00 00 00 00 00 0d 00 00 53
1070 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 bb
1080 00 0c 00 00 00 00 00 00 10 a2 00 1a 00 5a 00 10 3a
1090 00 10 00 04 00 01 00 01 00 00 10 8e 00 00 00 00 51
10a0 00 00 00 00 00 00 00 00 00 00 00 00 00 0e 00 00 70
10b0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 2f
10c0 00 0d 00 00 00 00 00 00 10 e2 00 1a 00 6a 00 10 57
10d0 00 10 00 04 00 01 00 01 00 00 10 ce 00 00 00 00 05
10e0 00 00 00 00 00 00 00 00 00 00 00 00 00 0f 00 00 8d
10f0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 a3
1100 00 0e 00 00 00 00 00 00 11 22 00 1a 00 7a 00 10 c2
1110 00 10 00 04 00 01 00 01 00 00 11 0e 00 00 00 00 ff
1120 00 00 00 00 00 00 00 00 00 00 00 00 00 10 00 00 93
1130 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 00
1140 00 0f 00 00 00 00 00 00 11 62 00 2a 00 0a 00 10 6f
1150 00 10 00 04 00 01 00 01 00 00 11 4e 00 00 00 00 b3
1160 00 00 00 00 00 00 00 00 00 00 00 00 00 11 00 00 b0
1170 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 74
1180 00 10 00 00 00 00 00 00 11 a2 00 2a 00 1a 00 10 8c
1190 00 10 00 04 00 01 00 01 00 00 11 8e 00 00 00 00 67
11a0 00 00 00 00 00 00 00 00 00 00 00 00 00 12 00 00 cd
11b0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 e8
11c0 00 11 00 00 00 00 00 00 11 e2 00 2a 00 2a 00 10 a9
11d0 00 10 00 04 00 01 00 01 00 00 11 ce 00 00 00 00 1b
11e0 00 00 00 00 00 00 00 00 00 00 00 00 00 13 00 00 ea
11f0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 5c
1200 00 12 00 00 00 00 00 00 12 22 00 2a 00 3a 00 10 14
1210 00 10 00 04 00 01 00 01 00 00 12 0e 00 00 00 00 15
1220 00 00 00 00 00 00 00 00 00 00 00 00 00 14 00 00 f0
1230 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 b9
1240 00 13 00 00 00 00 00 00 12 62 00 2a 00 4a 00 10 31
1250 00 10 00 04 00 01 00 01 00 00 12 4e 00 00 00 00 c9
1260 00 00 00 00 00 00 00 00 00 00 00 00 00 15 00 00 0d
1270 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 2d
1280 00 14 00 00 00 00 00 00 12 a2 00 2a 00 5a 00 10 4e
1290 00 10 00 04 00 01 00 01 00 00 12 8e 00 00 00 00 7d
12a0 00 00 00 00 00 00 00 00 00 00 00 00 00 16 00 00 2a
12b0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 a1
12c0 00 15 00 00 00 00 00 00 12 e2 00 2a 00 6a 00 10 6b
12d0 00 10 00 04 00 01 00 01 00 00 12 ce 00 00 00 00 31
12e0 00 00 00 00 00 00 00 00 00 00 00 00 00 17 00 00 47
12f0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 15
1300 00 16 00 00 00 00 00 00 13 22 00 2a 00 7a 00 10 d6
1310 00 10 00 04 00 01 00 01 00 00 13 0e 00 00 00 00 2b
1320 00 00 00 00 00 00 00 00 00 00 00 00 00 18 00 00 4d
1330 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 72
1340 00 17 00 00 00 00 00 00 13 62 00 3a 00 0a 00 10 83
1350 00 10 00 04 00 01 00 01 00 00 13 4e 00 00 00 00 df
1360 00 00 00 00 00 00 00 00 00 00 00 00 00 19 00 00 6a
1370 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 e6
1380 00 18 00 00 00 00 00 00 13 a2 00 3a 00 1a 00 10 a0
1390 00 10 00 04 00 01 00 01 00 00 13 8e 00 00 00 00 93
13a0 00 00 00 00 00 00 00 00 00 00 00 00 00 1a 00 00 87
13b0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 5a
13c0 00 19 00 00 00 00 00 00 13 e2 00 3a 00 2a 00 10 bd
13d0 00 10 00 04 00 01 00 01 00 00 13 ce 00 00 00 00 47
13e0 00 00 00 00 00 00 00 00 00 00 00 00 00 1b 00 00 a4
13f0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 ce
1400 00 1a 00 00 00 00 00 00 14 22 00 3a 00 3a 00 10 28
1410 00 10 00 04 00 01 00 01 00 00 14 0e 00 00 00 00 41
1420 00 00 00 00 00 00 00 00 00 00 00 00 00 1c 00 00 aa
1430 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 2b
1440 00 1b 00 00 00 00 00 00 14 62 00 3a 00 4a 00 10 45
1450 00 10 00 04 00 01 00 01 00 00 14 4e 00 00 00 00 f5
1460 00 00 00 00 00 00 00 00 00 00 00 00 00 1d 00 00 c7
1470 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 9f
1480 00 1c 00 00 00 00 00 00 14 a2 00 3a 00 5a 00 10 62
1490 00 10 00 04 00 01 00 01 00 00 14 8e 00 00 00 00 a9
14a0 00 00 00 00 00 00 00 00 00 00 00 00 00 1e 00 00 e4
14b0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 13
14c0 00 1d 00 00 00 00 00 00 14 e2 00 3a 00 6a 00 10 7f
14d0 00 10 00 04 00 01 00 01 00 00 14 ce 00 00 00 00 5d
14e0 00 00 00 00 00 00 00 00 00 00 00 00 00 1f 00 00 01
14f0 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 87
1500 00 1e 00 00 00 00 00 00 00 00 00 3a 00 7a 00 10 0f
1510 00 10 00 04 00 01 00 01 00 00 15 0e 00 00 00 00 57
1520 00 00 00 00 00 00 00 00 00 00 00 00 00 20 00 00 07
1530 00 00 00 00 00 00 00 10 00 10 00 00 00 00 00 00 e4
1540 00 1f 00 00 00 00 00 01 01 00 00 06 00 06 00 00 ca
1550 00 00 00 00 00 00 00 00 00 00 43 61 6e 20 6e 6f 71
1560 74 20 6f 70 65 6e 20 6f 75 74 70 75 74 2d 66 69 da
1570 6c 65 20 21 21 21 00 00 00 00 15 9c 00 30 00 a5 f2
1580 00 40 00 11 00 00 00 01 00 01 00 00 16 bc 00 00 da
1590 00 00 00 00 15 80 00 00 00 00 00 00 00 00 00 00 4b
15a0 00 00 00 00 00 01 01 00 00 07 00 04 00 00 00 00 98
15b0 00 00 15 94 00 00 00 00 43 61 6e 63 65 6c 00 00 e7
15c0 00 00 15 e4 00 58 00 32 00 40 00 11 00 00 00 01 5a
15d0 00 01 00 00 16 bc 00 00 00 00 00 00 15 c8 00 00 2a
15e0 00 00 00 00 00 00 00 21 00 00 00 00 00 01 01 00 3a
15f0 00 07 00 04 00 00 00 00 00 00 15 dc 00 00 00 00 90
1600 20 53 61 76 65 20 00 00 00 00 16 2c 00 08 00 91 20
1610 00 40 00 11 00 00 00 01 00 01 00 00 16 bc 00 00 08
1620 00 00 00 00 16 10 00 00 00 00 00 00 00 00 00 22 e0
1630 00 00 00 00 00 01 01 00 00 07 00 04 00 00 00 00 c6
1640 00 00 16 24 00 00 00 00 20 53 77 61 70 20 00 00 13
1650 00 00 16 74 00 58 00 91 00 40 00 11 00 00 00 01 54
1660 00 01 00 00 16 bc 00 00 00 00 00 00 16 58 00 00 fd
1670 00 00 00 00 00 00 00 23 00 00 00 00 00 01 01 00 aa
1680 00 07 00 04 00 00 00 00 00 00 16 6c 00 00 00 00 ab
1690 44 65 6c 65 74 65 00 00 00 00 0d 22 00 58 00 78 89
16a0 00 40 00 11 00 00 00 01 00 01 00 00 16 bc 00 00 4d
16b0 00 00 00 00 16 a0 00 00 00 00 00 00 00 00 00 24 37
16c0 00 00 00 00 00 01 01 00 00 07 00 04 00 00 00 00 0b
16d0 00 00 16 b4 00 00 00 00 20 55 6e 64 6f 20 00 00 33
16e0 00 00 00 00 01 00 00 05 00 00 16 cc 00 00 16 e0 5e
16f0 00 00 00 00 00 40 00 00 00 40 00 10 00 00 00 10 f9
```

```
1700 00 00 00 00 00 00 00 00 01 00 00 05 00 00 16 f0 e7
1710 00 00 00 00 00 02 00 01 00 3e 00 01 00 3e 00 0f 0b
1720 00 02 00 0f 00 02 00 01 00 00 00 00 00 00 03 ec 30
1730 00 00 01 13 00 00 00 00 00 00 00 08 00 00 00 0e 6c
1740 00 00 00 18 00 00 00 1e 00 00 00 28 00 00 00 36 67
1750 00 00 00 44 00 00 00 52 00 00 00 60 00 00 00 c2 28
1760 00 00 00 cc 00 00 00 e8 00 00 00 f2 00 00 00 fc cf
1770 00 00 01 0a 00 00 01 20 00 00 01 26 00 00 01 2c fa
1780 00 00 01 42 00 00 01 48 00 00 01 52 00 00 01 60 57
1790 00 00 01 68 00 00 01 70 00 00 01 90 00 00 01 9e 7e
17a0 00 00 01 b8 00 00 01 ce 00 00 01 d4 00 00 01 ea 19
17b0 00 00 01 f0 00 00 01 f6 00 00 02 00 00 00 02 12 d4
17c0 00 00 02 1a 00 00 02 30 00 00 02 4e 00 00 02 54 cf
17d0 00 00 02 5a 00 00 02 70 00 00 02 76 00 00 02 88 88
17e0 00 00 02 8e 00 00 02 94 00 00 02 a6 00 00 02 ac d1
17f0 00 00 02 de 00 00 02 e4 00 00 02 f4 00 00 02 fa 4a
1800 00 00 03 04 00 00 03 0c 00 00 03 12 00 00 03 20 36
1810 00 00 03 32 00 00 03 3c 00 00 03 48 00 00 03 52 59
1820 00 00 03 5e 00 00 03 68 00 00 05 32 00 00 05 42 1c
1830 00 00 05 56 00 00 05 62 00 00 05 68 00 00 05 76 49
1840 00 00 05 80 00 00 05 86 00 00 05 90 00 00 05 96 1c
1850 00 00 05 9c 00 00 05 a2 00 00 05 a8 00 00 05 b6 29
1860 00 00 05 d0 00 00 05 d8 00 00 05 e4 00 00 05 ee a4
1870 00 00 06 08 00 00 06 16 00 00 06 2e 00 00 06 34 fb
1880 00 00 06 3a 00 00 06 42 00 00 06 48 00 00 06 4e 6a
1890 00 00 06 58 00 00 06 5e 00 00 06 68 00 00 06 72 65
18a0 00 00 06 7e 00 00 06 86 00 00 06 8e 00 00 06 a8 ec
18b0 00 00 06 c6 00 00 06 d0 00 00 06 e0 00 00 07 14 86
18c0 00 00 07 1a 00 00 07 24 00 00 07 30 00 00 07 3c 4a
18d0 00 00 07 42 00 00 07 74 00 00 07 80 00 00 07 86 39
18e0 00 00 07 92 00 00 07 9e 00 00 07 aa 00 00 07 b0 44
18f0 00 00 07 dc 00 00 07 e2 00 00 08 64 00 00 08 6a 0d
1900 00 00 08 74 00 00 08 8a 00 00 08 92 00 00 08 b4 15
1910 00 00 08 c0 00 00 08 cc 00 00 08 ec 00 00 08 f8 be
1920 00 00 09 06 00 00 09 12 00 00 09 28 00 00 09 52 f1
1930 00 00 09 5a 00 00 09 72 00 00 09 78 00 00 09 7e 3e
1940 00 00 09 8a 00 00 09 a8 00 00 09 b4 00 00 09 ba f9
1950 00 00 09 c6 00 00 09 ce 00 00 09 d4 00 00 09 e0 fe
1960 00 00 09 e6 00 00 09 f0 00 00 09 f6 00 00 0a 00 4c
1970 00 00 0a 06 00 00 0a 14 00 00 0a 1c 00 00 0a 24 1e
1980 00 00 0a 2e 00 00 0a 40 00 00 0a 58 00 00 0a 5e 05
1990 00 00 0a 70 00 00 0a 76 00 00 0a 80 00 00 0a 86 ca
19a0 00 00 0a 90 00 00 0a 96 00 00 0a a2 00 00 0a d8 fb
19b0 00 00 0b 02 00 00 0b 06 00 00 0b 0c 00 00 0b 12 f2
19c0 00 00 0b 1c 00 00 0b 24 00 00 0b 28 00 00 0b 2e 1f
19d0 00 00 0b 34 00 00 0b 52 00 00 0b 58 00 00 0b 5e 62
19e0 00 00 0b 64 00 00 0b 6c 00 00 0b 74 00 00 0b 7c 43
19f0 00 00 0b 8a 00 00 0b 90 00 00 0b a2 00 00 0b a8 c4
1a00 00 00 0b b2 00 00 0b ba 00 00 0b c0 00 00 0b cc 3e
1a10 00 00 0c 08 00 00 0c 14 00 00 0c 24 00 00 0c 68 ff
1a20 00 00 0c 86 00 00 0c 8e 00 00 0c c6 00 00 0d 22 bd
1a30 00 00 0d 34 00 00 0d 62 00 00 0d 74 00 00 0d a2 d1
1a40 00 00 0d b4 00 00 0d e2 00 00 0d f4 00 00 0e 22 3b
1a50 00 00 0e 34 00 00 0e 62 00 00 0e 74 00 00 0e a2 9f
1a60 00 00 0e b4 00 00 0e e2 00 00 0e f4 00 00 0f 22 09
1a70 00 00 0f 34 00 00 0f 62 00 00 0f 74 00 00 0f a2 6d
1a80 00 00 0f b4 00 00 0f e2 00 00 0f f4 00 00 10 22 d7
1a90 00 00 10 34 00 00 10 62 00 00 10 74 00 00 10 a2 3b
1aa0 00 00 10 b4 00 00 10 e2 00 00 10 f4 00 00 11 22 a5
1ab0 00 00 11 34 00 00 11 62 00 00 11 74 00 00 11 a2 09
1ac0 00 00 11 b4 00 00 11 e2 00 00 11 f4 00 00 12 22 73
1ad0 00 00 12 34 00 00 12 62 00 00 12 74 00 00 12 a2 d7
1ae0 00 00 12 b4 00 00 12 e2 00 00 12 f4 00 00 13 22 41
1af0 00 00 13 34 00 00 13 62 00 00 13 74 00 00 13 a2 a5
1b00 00 00 13 b4 00 00 13 e2 00 00 13 f4 00 00 14 22 f8
1b10 00 00 14 34 00 00 14 62 00 00 14 74 00 00 14 a2 5c
1b20 00 00 14 b4 00 00 14 f4 00 00 15 54 00 00 15 66 19
1b30 00 00 15 6e 00 00 15 8c 00 00 15 9c 00 00 15 ae 62
1b40 00 00 15 b6 00 00 15 d4 00 00 15 e4 00 00 15 f6 df
1b50 00 00 15 fe 00 00 16 1c 00 00 16 2c 00 00 16 3e 73
1b60 00 00 16 46 00 00 16 64 00 00 16 74 00 00 16 86 ed
1b70 00 00 16 8e 00 00 16 ac 00 00 16 c4 00 00 16 c8 ec
1b80 00 00 16 e8 00 00 00 00 00 00 03 f2 00 00 03 eb cc
1b90 00 00 00 01 00 00 03 f2                         72
```

Mit Nanomon sofort zum lauffähigen Programm

NewInstall – Kompatibilitäts-Kur durch Abmagern

„Wenn der Berg nicht zum Propheten kommen will, muß der Prophet zum Berge gehen." — Solange Softwarehersteller sich nicht dazu bequemen, Software zu schreiben, die mit Erweiterungen arbeitet, muß man sich eben auf andere Weise helfen.

Daß der Amiga anders zu programmieren ist, als ein PC, soll sich mittlerweile herumgesprochen haben. Dennoch mangelt es vielen Programmen, speziell Spielen, noch an elementaren Punkten, um auch nur annähernd amiga-freundlich genannt zu werden. Multitasking ist ohnehin Luxus: Unter den „Action-Games" scheint lediglich das Amoeba-Invaders auf Fish-Disk 120 von Kennern des Amiga-Betriebssystems unter Einhaltung der zugehörigen Stil-Richtlinien geschrieben worden zu sein.

Vielen Programmierern ist bis heute die besondere Bedeutung des CHIP-Memory nicht bekannt, andere nehmen es gelassen hin, daß ihr Lieblingsassembler (ohne Namen nennen zu wollen . . .) keinen Code erzeugen kann, der automatisches Laden in einen speziellen Speicher erzwingt (dem raubkopierten Gaul schaut man nicht . . .). Daß diesem Assembler auch Include-Files völlig unbekannt sind, verschlechtert die Situation im Sinne eines sauberen Quelltextes um weitere Größenordnungen.

Der Leidtragende ist, außer dem Programmierer, der sich — oft ohne es zu wissen — das Leben mit veralteten Tools, die meist nicht einmal mehr antiquarischen Wert besitzen, das Leben schwermacht, natürlich letzten Endes — wie immer — der Käufer, der nicht das bekommt, was er für sein Geld erwarten kann.

So sind sogar noch heute manche Programme nicht mit Speichererweiterungen lauffähig. Andere übertreiben mit dem Kopierschutzverfahren ein wenig und bieten drei verschiedene Versionen ihres Programmes, jeweils für den Amiga 500, 1000 und 2000, an. Bitte teilen Sie uns Ihre Erfahrungen mit, so daß wir diese an alle Leser weitergeben können. Muß man die Diskette nicht booten, so kann man sich eventuell mit „Krücken" wie „NoFastMem" behelfen. Dieses nützliche kleine Programm der Workbench-Diskette verhindert den Zugriff auf Speicher, der über die Grundkonfiguration von 512 KBytes CHIP-Memory hinausgeht und erlaubt so die Ausführung speziell alter Programme, die zu Zeiten teurer Speichererweiterungen geschrieben wurden.

Ein anderes Problem stellen oft externe Diskettenlaufwerke dar, deren reine Anwesenheit derart stört, daß ein Programm nicht selten nur in der Konfiguration mit einem Laufwerk lauffähig ist. Schuld daran ist meist der für die Laufwerke allozierte Speicherbereich für interne Datenbereiche und Puffer. So „schluckt" jedes angeschlossene Laufwerk knapp 23 Kilobytes RAM, fast alles davon ist wertvoller CHIP-Speicher.

Wem diese Symptome nicht unbekannt sind, der wird sicher sehnsüchtig auf Abhilfe gewartet haben, die ihm sein Lieblingsspiel auch nach Aufstieg auf 1 MB und externes Laufwerk lauffähig hält.

Gegen schlechte Programmiermanieren hilft nur das Abschalten nicht benötigter Systemunterstützungen

Einzige Voraussetzung für das Funktionieren des hier abgedruckten Patches ist, daß das Programm nicht auf den auf der Diskette befindlichen Bootblock angewiesen ist, also entweder einen Standard-Bootblock (erzeugt durch „Install") oder einen puren Vorspann, der keine Auswirkungen auf das eigentliche Programm hat, besitzt.

Wenden Sie dazu das Programm „Install2" — sofern möglich — auf eine Kopie der Programmdiskette an und booten Sie anschließend wie gewohnt.

Auf diese Weise wird nun der CHIP-Speicher etwaig vorhandenem FAST-Memory vorgezogen, wodurch speziell statische Daten beim Laden des Programms automatisch im CHIP-Speicher abgelegt werden, wo sie durch die Custom-Chips des Amiga erreicht werden können.

Außerdem werden alle Laufwerke, verständlicherweise mit Ausnahme von DF0:, mittels Software „abgeschaltet", so daß dem Programm mehr Speicher zur Verfügung steht.

Auf diese Weise sollten auch bescheiden programmierte Spiele aus Deutschland lauffähig werden. Ich will damit weder den Eindruck erwecken, daß alle deutschen Spielepro-

grammierer in die genannte Kategorie fallen, noch, daß alles aus dem Ausland perfekt ist.

Vielleicht ringt sich aber doch der eine oder andere dazu durch, in Zukunft das Problem an der Wurzel zu packen und sich etwa zu dem recht guten Assembler des Aztec-C-Compilers oder auch einem der zahlreichen Assembler der Public-Domain- und Shareware-Disketten durchzuringen.

Sollte das Programm übrigens nach Installation des neuen Bootblocks noch immer nicht lauffähig sein, so wissen Sie bestimmt, wessen Software Sie in Zukunft sicherlich nicht mehr kaufen werden . . .

„requires no more than 512 KB and internal drive“

(Ralph Babel)

Programmname: Install2
Sprache: C


Auf Leserdiskette


Überprüfen mit checksum

```
/*                                                      653
** Install2.c - give preference to MEMF_CHIP and
   deactivate external drives                           653
** Copyright (C) 1987 Ralph Babel, Falkenweg 3, D-6204
   Taunusstein                                          653
** all rights reserved - alle Rechte vorbehalten        653
**                                                      653
** 23-May-1987 created Install1.c                       653
** 07-Aug-1987 changed for new boot code                653
** 30-Mar-1988 cleanup                                  653
*/                                                      358
                                                        358
/*** included files ***/                                580
                                                        580
#include <exec/types.h>                                 916
#include <exec/io.h>                                    659
#include <exec/memory.h>                                867
#include <devices/bootblock.h>                          251
#include <devices/trackdisk.h>                          464
#include <libraries/dos.h>                              805
#include <libraries/dosextens.h>                        489
                                                        489
/*** external function references ***/                  375
                                                        375
APTR AllocMem(ULONG, ULONG);                            442
VOID FreeMem(APTR, ULONG);                              841
struct IOStdReq *CreateStdIO(struct MsgPort *);         957
VOID DeleteStdIO(struct IOStdReq *);                    315
BYTE OpenDevice(char *, LONG, struct IORequest *, ULONG); 474
VOID CloseDevice(struct IORequest *);                   710
BYTE DoIO(struct IORequest *);                          557
struct Task *FindTask(char *);                          975
VOID CopyMem(APTR, APTR, ULONG);                        638
                                                        638
VOID printf(char *, );                                  403
                                                        403
LONG toupper(LONG);                                     698
                                                        698
/*** constants ***/                                     704
                                                        704
#define ROOT 880 /* default AmigaDOS root block (3.5
   inch) */                                             964
                                                        964
/*** structures ***/                                    898
                                                        898
struct BootSectors                                      286
 {                                                      793
 struct BootBlock bs_BootBlock;                         883
 UBYTE bs_Data[BOOTSECTS * TD_SECTOR - sizeof(struct
   BootBlock)];                                         190
 };                                                     026
                                                        026
/*** the boot code ***/                                 568
                                                        568
UWORD BootCode[] =                                      817
 {                                                      504
 0x2F02, 0x4EAE, 0xFF7C, 0x226E, 0x0142, 0x6020,        625
 0x0829, 0x0001, 0x000F, 0x6716, 0x2F09, 0x4EAE,        667
 0xFF04, 0x225F, 0x137C, 0x000A, 0x0009, 0x41EE,        045
 0x0142, 0x4EAE, 0xFEF2, 0x2242, 0x2411, 0x66DC,        255
 0x4EAE, 0xFF76, 0x241F, 0x43FA, 0x0032, 0x4EAE,        938
 0xFE0E, 0x4A80, 0x6724, 0x2040, 0x41E8, 0x0034,        205
 0x70FF, 0x20C0, 0x20C0, 0x2080, 0x43FA, 0x0026,        086
 0x4EAE, 0xFFA0, 0x4A80, 0x670A, 0x2040, 0x2068,        343
 0x0016, 0x7000, 0x4E75, 0x70FF, 0x4E75, 0x6469,        915
 0x736B, 0x2E72, 0x6573, 0x6F75, 0x7263, 0x6500,        783
 0x646F, 0x732E, 0x6C69, 0x6272, 0x6172, 0x7900         899
 };                                                     015
                                                        015
/*** entry point ***/                                   741
                                                        741
LONG main(argc, argv)                                   932
LONG argc;                                              168
char *argv[];                                           119
 {                                                      670
 struct BootSectors *bs;                                584
 struct IOStdReq *iosr;                                 702
 struct Process *pr;                                    961
 ULONG checksum, precsum;                               623
 UWORD i;                                               824
 LONG result;                                           946
                                                        946
 result = RETURN_WARN; /* user error */                 090
                                                        090
 if(argc == 2                                           966
 && toupper(argv[1][0]) == 'D'                          955
 && toupper(argv[1][1]) == 'F'                          019
 && argv[1][2] >= '0' && argv[1][2] <= '3'              876
 && argv[1][3] == ':'                                   262
 && argv[1][4] == '\0')                                 424
  {                                                     539
  result = RETURN_FAIL; /* no resources */              399
                                                        399
  if((bs = (struct BootSectors *)                       331
   AllocMem(sizeof(struct BootSectors), MEMF_CHIP |
   MEMF_CLEAR)) != NULL)                                277
   {                                                    676
   pr = (struct Process *)FindTask(NULL);               617
   if((iosr = CreateStdIO(&pr->pr_MsgPort)) != NULL)    441
    {                                                   760
    result = RETURN_ERROR; /* disk error */             814
                                                        814
    if(OpenDevice(TD_NAME, argv[1][2] - '0', (struct
     IORequest *)iosr, 0) == 0)                         119
     {                                                  670
     *(ULONG *)bs->bs_BootBlock.bb_id = BBNAME_DOS;     799
     bs->bs_BootBlock.bb_dosblock = ROOT;               220
     CopyMem((APTR)BootCode, (APTR)bs->bs_Data,
      sizeof(BootCode));                                407
                                                        407
     checksum = 0;                                      550
     for(i = 0; i < 256; ++i)                           539
      {                                                 986
      precsum = checksum;                               280
      if((checksum += ((ULONG *)bs)[i]) < precsum)      434
       ++checksum;                                      782
      }                                                 051
                                                        051
     bs->bs_BootBlock.bb_chksum = 0xffffffff - checksum; 206
                                                        206
     iosr->io_Command = CMD_WRITE;                      898
     iosr->io_Length  = sizeof(struct BootSectors);     764
     iosr->io_Data    = (APTR)bs;                       358
     iosr->io_Offset  = 0;                              240
     if(DoIO((struct IORequest *)iosr) == 0)            540
      {                                                 999
      iosr->io_Command = CMD_UPDATE;                    671
      (void)DoIO((struct IORequest *)iosr);             175
      }                                                 352
     if(iosr->io_Error != 0)                            867
      {                                                 154
      printf("%s failed, trackdisk error %ld\n", argv[0],
       iosr->io_Error);                                 460
      }                                                 009
     else                                               130
      {                                                 813
      result = RETURN_OK; /* everything fine */         287
      }                                                 808
     iosr->io_Command = TD_MOTOR;                       878
     iosr->io_Length  = 0;                              799
     (void)DoIO((struct IORequest *)iosr);              839
     CloseDevice((struct IORequest *)iosr);             659
     }                                                  500
    else                                                925
     {                                                  908
     printf("Unable to open trackdisk device, error
      %ld\n", iosr->io_Error);                          521
     }                                                  754
    DeleteStdIO(iosr);                                  613
    }                                                   950
   else                                                 127
    {                                                   774
    printf("Unable to create IO request\n");            066
    }                                                   983
   }                                                    616
  else                                                  337
   {                                                    408
   printf("Insufficient free store\n");                 419
   }                                                    476
  }                                                     169
 else                                                   802
  {                                                     309
  printf("Usage: %s [DF0:|DF1:|DF2:|DF3:]\n", argv[0]); 715
  }                                                     228
                                                        228
 return result;                                         566
 }                                                      339
```

Install2 verhilft dem Rechner zu Diät.

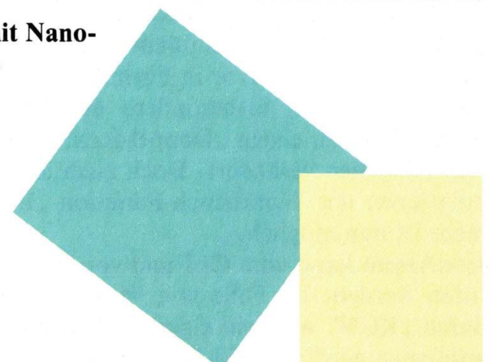
```
;                                                        059
;Install2.bat                                            248
;                                                        235
FailAt 1                                                 421
LC1 -b0 -ceft -oT: Install2                              226
LC2 -r1 -s -v T:Install2                                 445
BLink LIB:Astartup.obj+T:Install2.o TO Install2 LIB
  LIB:amiga.lib+LIB:lcnb.lib SC SD ND NOALVS BATCH       026
Delete T:Install2.o                                      601
FailAt 10                                                455
```

Batchfile zur Montage mit Lattice-C-4.00

```
*                                                        042
* BootCode2.asm - give preference to
  MEMF_CHIP and deactivate external drives               591
* Copyright (C) 1987 Ralph Babel,
  Falkenweg 3, D-6204 Taunusstein                        599
* all rights reserved - alle Rechte
  vorbehalten                                            607
*                                                        789
* 07-Aug-1987 created                                    250
* 30-Mar-1988 cleanup                                    291
*                                                        777
                                                         777
        nolist                                           486
                                                         486
        include "exec/types.i"                           170
        include "exec/execbase.i"                        633
        include "exec/memory.i"                          749
        include "exec/resident.i"                        494
        include "devices/bootblock.i"                    358
        include "resources/disk.i"                       438
                                                         438
        list                                             166
                                                         166
*                                                        152
* registers                                              194
*                                                        516
                                                         516
REG_SysBase     equr    a6                               807
                                                         807
*                                                        293
* macros                                                 528
*                                                        762
                                                         762
xlib    macro                                            779
        xref    _LVO\1                                   141
        endm                                             965
                                                         965
callsys macro                                            554
        jsr     _LVO\1(REG_SysBase)                      654
        endm                                             366
                                                         366
*                                                        704
* IMPORT, external symbol references                     944
*                                                        026
                                                         026
        xlib    Forbid                                   531
        xlib    Permit                                   235
        xlib    Remove                                   840
        xlib    Enqueue                                  939
        xlib    OpenResource                             404
        xlib    FindResident                             214
                                                         214
*                                                        776
* constants                                              819
*                                                        449
                                                         449
ROOT    equ     880     ;default AmigaDOS
  root block (3.5 inch)                                  549
NEW_PRI equ     10      ;priority for
  MEMF_CHIP regions                                      680
                                                         680
*                                                        690
* code section                                           320
*                                                        106
                                                         106
        section "text",code                              499
                                                         499
*                                                        385
* magic constants                                        921
*                                                        775
                                                         775
        BBID_DOS                                         841
        dc.l    0       ;checksum                        440
        dc.l    ROOT                                     285
                                                         285
*                                                        699
* boot code, called on system start-up                   759
*                                                        717
* In:                                                    252
*                                                        270
* A1: IOStdReq for TD_NAME, UNIT_0                       589
* A6: SysBase                                            550
*                                                        048
* Out:                                                   360
*                                                        626
* A0: start address to jump to (D0 ==
  zero)                                                  672
* D0: failure code, null if successful                   557
*                                                        139
                                                         139
        move.l  d2,-(sp)                                 187
                                                         187
        callsys Forbid                                   396
                                                         396
        movea.l
         LH_HEAD+MemList(REG_SysBase),a1                 303
        bra.s   3$                                       584
                                                         584
1$      btst.b
  #MEMB_CHIP,MH_ATTRIBUTES+1(a1)  ;hi ..
  lo                                                     742
        beq.s   2$                                       317
                                                         317
        move.l  a1,-(sp)                                 821
        callsys Remove                                   022
        movea.l (sp)+,a1                                 909
                                                         909
        move.b  #NEW_PRI,LN_PRI(a1)                      602
        lea     MemList(REG_SysBase),a0                  952
        callsys Enqueue                                  731
                                                         731
2$      movea.l d2,a1                                    843
                                                         843
3$      move.l  (a1),d2 ;move.l
  LN_SUCC(a1),d2                                         784
        bne.s   1$                                       679
                                                         679
        callsys Permit                                   483
                                                         483
        move.l  (sp)+,d2                                 357
                                                         357
        lea     diskName(pc),a1                          397
        callsys OpenResource                             450
        tst.l   d0                                       083
        beq.s   error                                    466
                                                         466
        movea.l d0,a0                                    985
                                                         985
        lea     DR_UNITID+4(a0),a0                       639
        moveq   #DRT_EMPTY,d0                            486
                                                         486
        move.l  d0,(a0)+        ;unit 1                  680
        move.l  d0,(a0)+        ;unit 2                  923
        move.l  d0,(a0)         ;unit 3                  664
                                                         664
        lea     dosName(pc),a1                           603
        callsys FindResident                             309
        tst.l   d0                                       574
        beq.s   error                                    121
        movea.l d0,a0                                    484
        movea.l RT_INIT(a0),a0                           574
        moveq   #0,d0                                    497
        rts                                              734
                                                         734
error   moveq   #-1,d0                                   461
        rts                                              514
                                                         514
diskName:                                                152
        dc.b    'disk.resource',0                        150
                                                         150
dosName:                                                 211
        dc.b    'dos.library',0                          289
                                                         289
        end                                              220
```

Assembler pur: Das steht im neuen Bootblock.

```
000 00 00 03 f3 00 00 00 00 00 00 00 02 00 00 00 00 1c
010 00 00 00 01 00 00 01 94 00 00 00 ba 00 00 03 e9 39
020 00 00 01 94 23 cf 00 00 00 1c 23 c0 00 00 00 24 a8
030 23 c8 00 00 00 28 2c 79 00 00 00 04 23 ce 00 00 ac
040 00 04 93 c9 4e ae fe da 28 40 4a ac 00 ac 67 00 b9
050 00 f6 61 00 01 8c 91 c8 20 2c 00 ac e5 88 20 30 d3
060 08 10 e5 88 48 e7 00 30 45 f9 00 00 00 ac 47 f9 00
070 00 00 00 2c 20 40 70 00 10 18 42 30 08 00 26 c8 b7
080 20 39 00 00 00 24 20 79 00 00 00 28 43 f0 08 00 d5
090 0c 21 00 20 52 c8 ff fa 42 29 00 01 12 18 67 5c 4e
0a0 0c 01 00 20 67 f6 0c 01 00 09 67 f0 26 ca 0c 01 7e
0b0 00 22 67 14 14 c1 12 18 67 42 0c 01 00 20 67 04 00
0c0 14 c1 60 f2 42 1a 60 d4 12 18 67 30 0c 01 00 22 a7
0d0 67 f2 0c 01 00 2a 66 20 12 18 0c 01 00 4e 67 06 91
0e0 0c 01 00 6e 66 04 72 0a 60 0e 0c 01 00 45 67 06 88
0f0 0c 01 00 65 66 02 72 1b 14 c1 60 cc 42 12 42 93 94
100 20 3c 00 00 00 28 90 8b 46 80 e4 88 4c df 0c 00 e1
110 48 79 00 00 00 2c 2f 00 4e b9 00 00 05 70 23 c0 35
120 00 00 00 10 4e b9 00 00 05 80 23 c0 00 00 00 14 ee
130 23 c0 00 00 00 18 4e b9 00 00 01 dc 70 00 2e 79 9e
140 00 00 00 1c 4e 75 61 00 00 98 61 00 00 82 23 c0 87
150 00 00 00 20 2f 00 42 a7 24 40 20 2a 00 24 67 10 73
160 2c 79 00 00 00 08 20 40 22 28 00 00 4e ae ff 82 87
170 4e b9 00 00 01 dc 70 00 60 04 20 2f 00 04 2e 79 b2
180 00 00 00 1c 2f 00 2c 79 00 00 00 04 20 39 00 00 92
190 00 08 67 06 22 40 4e ae fe 62 4a b9 00 00 00 20 f8
1a0 67 0e 4e ae ff 7c 22 79 00 00 00 20 4e ae fe 86 2a
1b0 20 1f 4e 75 48 e7 01 06 2e 3c 00 03 80 07 2c 78 5c
1c0 00 04 4e ae ff 94 4c df 60 80 70 64 60 b0 41 ec f4
1d0 00 5c 4e ae fe 80 41 ec 00 5c 4e ae fe 8c 4e 75 4e
1e0 43 fa 00 12 70 00 4e ae fd d8 23 c0 00 00 00 08 36
1f0 67 c2 4e 75 64 6f 73 2e 6c 69 62 72 61 72 79 00 a1
200 4e 55 ff e6 48 e7 20 00 70 05 2b 40 ff e6 0c ad 2b
210 00 00 00 02 00 08 66 00 02 3c 22 6d 00 0c 20 69 99
220 00 04 10 10 48 80 48 c0 2f 00 4e ba 04 30 58 4f 46
230 0c 80 00 00 00 44 66 00 02 1c 22 6d 00 0c 20 69 a9
240 00 04 52 88 10 10 48 80 48 c0 2f 00 4e ba 04 0e 11
250 58 4f 0c 80 00 00 00 46 66 00 01 fa 22 6d 00 0c 54
260 20 69 00 04 22 48 54 89 10 11 0c 00 00 30 6d 00 32
270 01 e4 22 48 54 89 10 11 0c 00 00 33 6e 00 01 d6 26
280 22 48 56 89 10 11 0c 00 00 3a 66 00 01 c8 58 88 1d
290 10 10 4a 00 66 00 01 be 70 14 2b 40 ff e6 2f 3c 65
2a0 00 01 00 02 48 78 04 00 4e ba 03 0a 50 4f 2b 40 3a
2b0 ff fc 4a 80 67 00 01 90 42 a7 4e ba 03 28 58 4f 91
2c0 20 40 2b 48 ff f4 d0 fc 00 5c 2f 08 4e ba 02 12 d3
2d0 58 4f 2b 40 ff f8 4a 80 67 00 01 5e 72 0a 2b 41 40
2e0 ff e6 22 6d 00 0c 20 69 00 04 54 88 12 10 48 81 a8
2f0 48 c1 04 81 00 00 00 30 42 a7 2f 00 2f 01 48 79 2c
300 00 00 02 30 4e ba 02 f2 4f ef 00 10 4a 00 66 00 43
310 01 02 20 6d ff fc 20 bc 44 4f 53 00 21 7c 00 00 e2
320 03 70 00 08 d0 fc 00 0c 48 78 00 84 2f 08 48 79 8c
330 00 00 01 ac 4e ba 03 0a 4f ef 00 0c 42 ad ff f0 dc
340 42 6d ff ea 32 2d ff ea 0c 41 01 00 64 2c 20 2d 66
350 ff f0 24 01 48 c2 e5 82 20 6d ff fc 22 30 28 00 af
360 d3 ad ff f0 2b 40 ff ec 24 2d ff f0 b4 80 64 04 9e
370 52 ad ff f0 52 6d ff ea 60 ca 70 ff 90 ad ff f0 d5
380 20 6d ff fc 21 40 00 04 22 6d ff f8 33 7c 00 03 c8
390 00 1c 23 7c 00 00 04 00 00 24 23 48 00 28 42 a9 21
3a0 00 2c 2f 09 4e ba 02 86 58 4f 4a 00 66 12 20 6d 93
3b0 ff f8 31 7c 00 04 00 1c 2f 08 4e ba 02 70 58 4f 2e
3c0 20 6d ff f8 4a 28 00 1f 67 20 10 28 00 1f 48 80 9a
3d0 48 c0 2f 00 22 6d 00 0c 2f 11 48 79 00 00 02 42 8b
3e0 4e ba 00 e2 4f ef 00 0c 60 06 70 00 2b 40 ff e6 47
3f0 20 6d ff f8 31 7c 00 09 00 1c 42 a8 00 24 2f 08 55
400 4e ba 02 2a 58 4f 2f 2d ff f8 4e ba 02 0c 58 4f 47
410 60 1a 20 6d ff f8 10 28 00 1f 48 80 48 c0 2f 00 bd
420 48 79 00 00 02 62 4e ba 00 9c 50 4f 2f 2d ff f8 29
430 4e ba 00 c2 58 4f 60 2e 48 79 00 00 02 8e 4e ba cb
440 00 84 58 4f 60 20 48 79 00 00 02 ac 4e ba 00 76 e0
450 58 4f 60 12 20 6d 00 0c 2f 10 48 79 00 00 02 c6 23
460 4e ba 00 62 50 4f 20 2d ff e6 4c df 00 04 4e 5d bb
470 4e 75 00 00 48 e7 00 3a 28 6f 00 14 20 6f 00 18 35
480 22 6f 00 1c 45 fa 00 3a 4f ef ff 74 26 4f 2c 79 29
490 00 00 00 04 4e ae fd f6 70 ff 4a 1b 57 c8 ff fc 9e
4a0 46 80 67 12 2f 00 48 6f 00 04 48 54 4e b9 00 00 7a
4b0 05 54 4f ef 00 0c 4f ef 00 8c 4c df 5c 00 4e 75 2e
4c0 16 c0 4e 75 20 2f 00 04 48 6f 00 08 2f 00 2f 39 92
4d0 00 00 00 14 4e b9 00 00 04 50 4f ef 00 0c 4e 75 a5
4e0 20 2f 00 04 48 78 00 30 2f 00 4e b9 00 00 04 e0 43
4f0 50 8f 4e 75 20 2f 00 04 2f 00 4e b9 00 00 05 22 31
500 58 8f 4e 75 48 e7 38 00 24 2f 00 10 26 2f 00 14 1a
510 4a 82 66 04 70 00 60 28 2f 3c 00 01 00 01 2f 03 ef
520 4e b9 00 00 05 90 20 40 28 08 50 8f 66 02 60 e4 ba
530 11 7c 00 05 00 08 31 43 00 12 21 42 00 0e 20 08 e9
540 4c df 00 1c 4e 75 20 6f 00 04 20 08 66 02 60 24 a2
550 11 7c 00 ff 00 08 70 ff 21 40 00 14 70 ff 21 40 12
560 00 18 70 00 30 28 00 12 2f 00 2f 08 4e b9 00 00 2a
570 05 a8 50 8f 4e 75 00 00 48 e7 30 02 2c 79 00 00 01
580 00 08 4c ef 00 0e 00 10 4e ae ff d0 4c df 40 0c 6c
590 4e 75 00 00 2f 0e 2c 79 00 00 00 08 4e ae ff ca 2c
5a0 2c 5f 4e 75 2f 0e 2c 79 00 00 00 08 4e ae ff c4 2e
5b0 2c 5f 4e 75 2f 0e 2c 79 00 00 00 04 4c ef 00 03 52
5c0 00 08 4e ae ff 3a 2c 5f 4e 75 00 00 2f 0e 2c 79 16
5d0 00 00 00 04 22 6f 00 08 20 2f 00 0c 4e ae ff 2e 1b
5e0 2c 5f 4e 75 2f 0e 2c 79 00 00 00 04 22 6f 00 08 7c
5f0 4e ae fe da 2c 5f 4e 75 2f 0e 2c 79 00 00 00 04 84
600 20 6f 00 08 4c ef 02 01 00 0c 22 2f 00 14 4e ae f0
610 fe 44 2c 5f 4e 75 00 00 2f 0e 2c 79 00 00 00 04 ad
620 22 6f 00 08 4e ae fe 3e 2c 5f 4e 75 2f 0e 2c 79 ad
630 00 00 00 04 22 6f 00 08 4e ae fe 38 2c 5f 4e 75 aa
640 2f 0e 2c 79 00 00 00 04 4c ef 03 00 00 08 20 2f 1d
650 00 10 4e ae fd 90 2c 5f 4e 75 00 00 20 2f 00 04 85
660 0c 00 00 61 6d 0a 0c 00 00 7a 6e 04 04 00 00 20 30
670 4e 75 00 00 00 03 ec 00 00 00 0a 00 00 00 00 b1
680 00 00 04 fe 00 00 05 4a 00 00 04 c8 00 00 04 d8 83
690 00 00 04 b2 00 00 04 8a 00 00 00 f6 00 00 01 02 84
6a0 00 00 01 4e 00 00 01 14 00 00 00 27 00 00 00 01 3c
6b0 00 00 06 20 00 00 06 0c 00 00 05 f8 00 00 05 d8 5f
6c0 00 00 05 c4 00 00 05 ac 00 00 05 94 00 00 05 84 1e
6d0 00 00 05 74 00 00 05 5a 00 00 04 ac 00 00 04 38 7b
6e0 00 00 04 24 00 00 04 16 00 00 03 fe 00 00 03 b8 a2
6f0 00 00 03 0c 00 00 02 dc 00 00 01 c8 00 00 01 84 f8
700 00 00 01 78 00 00 01 6a 00 00 01 5c 00 00 01 3e 1f
710 00 00 01 2c 00 00 01 1c 00 00 01 0e 00 00 01 08 7e
720 00 00 00 fc 00 00 00 ee 00 00 00 de 00 00 00 64 d5
730 00 00 00 5e 00 00 00 4c 00 00 00 46 00 00 00 1a b0
740 00 00 00 0e 00 00 00 08 00 00 00 02 00 00 00 00 5b
750 00 00 03 f2 00 00 03 ea 00 00 00 ba 00 21 00 01 7e
760 00 00 00 00 00 00 00 00 00 00 00 00 ff ff ff ff f1
770 ff ff ff ff ff ff ff ff 00 00 00 00 00 00 00 00 42
780 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 f7
790 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 54
7a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 b1
7b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0e
7c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 6b
7d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 c8
7e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 25
7f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 82
800 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 c8
810 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 25
820 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 82
830 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 df
840 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 3c
850 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 99
860 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 f6
870 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 53
880 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 b0
890 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0d
8a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 6a
8b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 c7
8c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 24
8d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 81
8e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 de
8f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 3b
900 00 00 00 00 00 00 00 00 00 2f 02 4e ae ff 7c 22 6e d1
910 01 42 60 20 08 29 00 01 00 0f 67 16 2f 09 4e ae 77
920 ff 04 22 5f 13 7c 00 0a 00 09 41 ee 01 42 4e ae 2f
930 fe f2 22 42 24 11 66 dc 4e ae ff 76 24 1f 43 fa cc
940 00 32 4e ae fe 0e 4a 80 67 24 20 40 41 e8 00 34 89
950 70 ff 20 c0 20 c0 20 80 43 fa 00 26 4e ae ff a0 ff
960 4a 80 67 0a 20 40 20 68 00 16 70 00 4e 75 70 ff 26
970 4e 75 64 69 73 6b 2e 72 65 73 6f 75 72 63 65 00 28
980 64 6f 73 2e 6c 69 62 72 61 72 79 00 74 72 61 63 54
990 6b 64 69 73 6b 2e 64 65 76 69 63 65 00 00 25 73 b6
9a0 20 66 61 69 6c 65 64 2c 20 74 72 61 63 6b 64 69 26
9b0 73 6b 20 65 72 72 6f 72 20 25 6c 64 0a 00 55 6e 86
9c0 61 62 6c 65 20 74 6f 20 6f 70 65 6e 20 74 72 61 55
9d0 63 6b 64 69 73 6b 2e 64 65 76 69 63 65 2c 20 65 d0
9e0 72 72 6f 72 20 25 6c 64 0a 00 55 6e 61 62 6c 65 ae
9f0 20 74 6f 20 63 72 65 61 74 65 20 49 4f 20 72 65 82
a00 71 75 65 73 74 0a 00 00 49 6e 73 75 66 66 69 63 91
a10 69 65 6e 74 20 66 72 65 65 20 73 74 6f 72 65 0a 6c
a20 00 00 55 73 61 67 65 3a 20 25 73 20 7b 44 46 30 a4
a30 3a 7c 44 46 31 3a 7c 44 46 32 3a 7c 44 46 33 3a 11
a40 7d 0a 00 00 00 00 03 f2                         92
```



Nach Eingabe mit Nanomon ergibt diese Hex-Liste das lauffähige Programm

2000 Mark für das



Ikonen, komplett manipuliert

Icons enthalten mehr Informationen als nur Grafikdaten. IconAssem ermöglicht, was mit IconEd der Workbench-Diskette nicht machbar ist: Manipulation der Systemdaten, die in den .info-Files versteckt sind. Dieses aufwendige Programm, das die Features einiger Einzelprogramme zusammenfügt und erweitert, wird mit 2000 Mark honoriert.



Grafische Vielfalt für die Workbench nutzen: IconAssem macht es möglich

Sicherlich ist es Ihnen auch schon einmal passiert, daß Sie auf einer Public-Domain-Diskette ein Piktogramm entdeckt haben, das genau zu Ihrer Anwendung passen würde, wäre es nur vom richtigen Typ. Man kann davon ausgehen, daß Murphy seine Finger im Spiel hatte, als die Version 33.59 der Workbench-Diskette zusammengestellt wurde: Hier nämlich war der Mülleimer (neues Design) nicht mehr vom Typ GARBAGE (siehe Workbench-Menüpunkt „Info"), sondern versehentlich eine Schublade (DRAWER), entsprechend etwa „Utilities" oder „Empty". So konnte plötzlich der Mülleimer nicht mehr geleert, dafür aber gelöscht werden. Folge: heillose Verwirrung bei den Anwendern.

IconAssem bietet die Kombination all der Features, die, wenigstens teilweise, zuvor in Form vieler kleiner Programme geboten wurden, insbesondere die Verknüpfung zweier Icon-Images zu einem „Doppel-Icon", das beim Anklicken sein Aussehen verändert. Doch auch icon-bezogene Manipulationen der Workbench-Funktion „Info" und weiteres mehr ist nun möglich.

IconAssem kann vom CLI und von Workbench aus aufgerufen werden. Im Falle von Workbench oder beim Start durch „RUN" wird ein eigenes Console-Window eröffnet, ansonsten wird das CLI-Fenster genutzt. Über die Gründe hierfür können Sie im Artikel „RunBack — Hintergründiges mit dem Amiga", nachlesen. Nun können, ähnlich NanoMon, Anweisungen eingegeben werden, die dann von IconAssem ausgeführt werden.

Kenntnis über den internen Aufbau eines Icon wird im folgenden vorausgesetzt. Machen Sie sich mit den Headerfiles <workbench/*.h> und dem zugehörigen Kapitel des zweiten ROM Kernel Manuals vertraut.

Die Befehle lauten:

CD [<DirName>]

CD — Diese Anweisung entspricht vollständig dem gleichnamigen CLI-Befehl. Folgt kein Parameter, so wird der Pfadname des aktuellen Verzeichnisses ausgegeben. Beim Start von IconAssem wird das aktuelle Verzeichnis vom CLI oder von Workbench übernommen. Änderungen gelten jedoch nur innerhalb von IconAssem und nicht mehr nach dem Verlassen des Programms, Sie bleiben also im zuvor vom CLI gewählten „Current Directory".

Dir [<DirName>]

DIR — Dieser Befehl gibt alle Verzeichnisse sowie alle Icon-Dateien aus, letztere aber OHNE die Endung „.info". Andere Dateien werden der Übersichtlichkeit halber ignoriert,

da eine Bearbeitung von IconAssem ohnehin nicht möglich ist. Bei fehlendem Verzeichnisnamen als Parameter wird vom aktuellen Arbeitsverzeichnis, das mit CD bestimmt wird, ausgegangen.

Save <IconName>

SAVE — Hierdurch werden die gegenwärtigen Daten als Icon auf Diskette unter dem spezifizierten Namen abgelegt, bei fehlendem Gerätenamen relativ zum aktuellen Verzeichnis. Ein Icon kann nur gespeichert werden, wenn zumindest das erste Image definiert ist, anderweitige Versuche sind strafbar und werden mit Fehlermeldung zurückgewiesen.

GetData <IconName>

GETDATA — Hierdurch werden alle Icon-Daten (außer den Images) geladen, so etwa der Typ, das ToolWindow und weiteres mehr, je nach Typ des Icons. Eine Sonderbehandlung erfährt das ToolTypes-Array, das nur dann geladen wird, wenn momentan keine ToolTypes definiert sind. Hierdurch wird versehentliches Überschreiben verhindert. Wie auch in allen anderen Fällen, in denen ein Icon-Name als Parameter spezifiziert wird, so muß dies OHNE das Suffix „.info“ geschehen, da dieses automatisch vom Betriebssystem angehängt wird.

GetImage <DstImage> [<IconName> [<SrcImage>]]

GETIMAGE — Dies ist die zugehörige Anweisung, die sich mit den Bilddaten eines Icon befaßt. Dabei muß angegeben werden, ob das zu ladende Bild als „normales“ Bild gelesen werden soll (DestImage = 0, fehlt der Parameter wird 0 angenommen), oder ob es als Aktivdarstellung (zweites Bild des Doppelicon: DstImage = 1), die beim Anklicken des Icon angezeigt wird, geladen werden soll. Auch muß natürlich ausgewählt werden, ob das erste oder zweite Image des auf Diskette befindlichen Icons zu extrahieren ist (SrcImage = 0 oder 1), auf diese Weise lassen sich auch die beiden Images eines Icons vertauschen. Werden die dem Zielparameter folgenden Daten ausgelassen, so wird das bezeichnete Image gelöscht. Dies muß dann geschehen, wenn aus einem Doppel-Icon ein einfaches gemacht werden soll. Für das Löschen des ersten Images gibt es eigentlich keinen Grund, da das Laden eines neues Icons das Löschen der alten Grafik impliziert und ein Icon ohne Bild ohnehin nicht gespeichert werden kann. Die beiden Images eines Doppel-Icons sollten die gleiche Größe besitzen, da es sonst zu konfuser Bildschirmdarstellung kommen kann.

IconType [DISK¦DRAWER¦TOOL¦PROJECT¦GARBAGE]

ICONTYPE — Fünf verschiedene Icontypen sind definiert: DISK, DRAWER, TOOL, PROJECT und GARBAGE. Wie aus der Workbench-Menüfunktion „Info“ ersichtlich, besitzen dabei alle Icons das ToolType-Array als Attribut. PROJECT und DISK besitzen beide das DefaultTool. Bei PROJECT bezeichnet es das Tool, das bei Eröffnung dieses Projekts automatisch gestartet wird, sofern das Project-Icon nicht durch Extended Selection einem anderen Programm übergeben wird. Bei Disketten ist das DefaultTool üblicherweise „SYS:System/DiskCopy“ und bezeichnet das Kopierprogramm, das beim Übereinanderziehen von Disketten-Icons in Aktion tritt.
Nur bei schubladenartigen Icons, DISK, DRAWER und GARBAGE also, lassen sich die Daten eines dazugehörigen Fensters modifizieren.
Für TOOLS und PROJECTS gibt es dann noch die Größe des Stacks, der dem zu startenden Tool übergeben wird. Ein „Stack, der nach der LIFO-Methode arbeitet“ ist übrigens ein Pleonasmus, aber das sei nur nebenbei bemerkt . . .
Das ToolWindow, ebenfalls für TOOLS und PROJECTS, soll eigentlich die Daten des Fensters festlegen, das selbsttätig nach Start des Programms eröffnet wird. Dieses Feature ist jedoch nicht auf die dokumentierte Weise funktionsfähig und wird daher nur von den wenigsten Programmen genutzt.

IconSize [(<Width> <Height>) ¦ ADJUST]

ICONSIZE — Die Größe eines Icons kann hierdurch bestimmt werden. Dies definiert den Bereich, der mit der Maus angeklickt werden kann und muß nicht mit der Größe der Images identisch sein. Durch Angabe von „ADJUST“ kann jedoch diese Größe der maximalen Ausdehnung der Images angepaßt werden. Ohne Parameter werden die aktuellen Werte ausgegeben. Wenn Sie die vertikale Ausdehnung des Icons größer als die der Images wählen, so wird der zu dem Icon gehörige Datei- oder Verzeichnisname in so vielen Pixelreihen Abstand dargestellt, wie die Icongröße die der Images übersteigt.

IconPosition [(<XPosition> <YPosition>) ¦ NONE]

ICONPOSITION — Die Position eines Icons erfolgt relativ zum Fenster, in dem es sich befindet. Soll sich der Computer selbst eine Stelle aussuchen, so geben Sie einfach „NONE“ an.

HighLight [COMP¦BACKFILL]

HIGHLIGHT — „COMP“ bewirkt das Invertieren des Icons beim Anklicken, „BACKFILL“ eliminiert zusätzlich den „Ring“ oder „Rand“ in Farbe 3 (orange in der Standardeinstellung), der sich hierbei ergeben kann. Diese Option kann auch durch den IconEd gewählt werden. Der dritte Modus, „IMAGE“, wird implizit eingeschaltet, wenn Sie ein zweites Image einladen. Entsprechend wird dieses zweite Image gelöscht, wenn einer der beiden anderen Modi spezifiziert wird.

WindowSize [<Width> <Height>]

WINDOWSIZE — Ebenso wie die beiden folgenden Befehle, so kann auch „WindowSize“ nur auf Icons vom Typ DISK, DRAWER und GARBAGE angewandt werden. „WindowSize“ bezieht sich auf die Größe des zu öffnenden Fensters.

WindowPosition [<XPosition> <YPosition>]

WINDOWPOSITION — Durch dieses Kommando läßt sich die Position des Fensters auf dem Workbench-Screen, genauer genommen dem Backdrop-Window, kontrollieren.

WindowOrigin [<XOrigin> <YOrigin>] ►

WINDOWORIGIN — Da sich der Inhalt eines Fensters mit Hilfe der beiden angeschlossenen Propgadgets verschieben läßt und nicht unbedingt alle Objekte gleichzeitig sichtbar sind, muß auch gespeichert werden, welcher Ausschnitt denn gerade sichtbar ist, ähnlich einer SUPERBITMAP, obwohl Workbench-Windows vom Typ SIMPLE_REFRESH sind. Den Umfang dieser Verschiebung gibt „WindowOrigin" an.

ToolType [(<String> [<Tool>]) ¦ (<Tool> DEL)]

TOOLTYPE — Dieses Feld von Strings, für deren Aufbau von Workbench keine Regeln auferlegt werden, ist für jeden Icontyp definiert. Manche Programme, so etwa „Notepad" oder „SetMap", verwenden es, um gewisse Optionen vom Anwender festsetzen zu lassen. Schlagen Sie hierzu im Handbuch nach. Bei nur einem Parameter wird dieser an die Liste der Tooltypes angehängt, andernfalls an die als zweiten Parameter angegebene Position eingefügt. Das Löschen von Einträgen geschieht durch Angabe der Nummer und des Schlüsselworts „DEL".

DefaultTool [<DefaultTool> ¦ NONE]

DEFAULTTOOL — Wie bereits oben beschrieben, wird hierdurch für PROJECTS und DISKS das DefaultTool festgelegt.

ToolWindow [<ToolWindow> ¦ NONE]

TOOLWINDOW — Sollte vom Typ „RAW:" oder „CON:" sein, die genaue Syntax für diese Fenster entnehmen Sie Ihrem DOS-Handbuch. Dieses Feature wird zur Zeit (Kickstart 1.3 und kleiner) aufgrund (vermutlich) eines Fehlers in Workbench nur selten genutzt.
STACKSIZE — Die Größe des Stacks für ein PROJECT oder das zugehörige TOOL. Der Wert null bewirkt, daß der Ersatzwert von (gegenwärtig) 4 Kilobytes stattdessen verwendet wird.

QUIT — Hierdurch können Sie IconAssem verlassen, etwa um mit weiteren Programmen zu arbeiten (wozu Sie aber, außer bei mangelndem Speicher, bei einem Multitasking-Rechner die Ausführung eines Programms nicht beenden müssen).
Durch Eingabe von „HELP" läßt sich jederzeit eine Tabelle aller Kommandos mit Syntaxbeschreibung aufrufen.
Die Fehlermeldungen sind weitestgehend selbsterklärend. Bei Fehlbenutzung wird die vom Programm erwartete Syntax angezeigt, „illegal quantity" zeigt an, daß ein numerisches Argument fehlerhaft spezifiziert wurde. AmigaDOS-Fehler werden im Klartext ausgegeben.
Durch Angabe von NEWCON anstelle von CON in „main()" („static char window[]") läßt sich dieses Programm übrigens auch mit einer Command-History betreiben. Beim Start ohne RUN von einer Shell mit NEWCON wird die History automatisch übernommen.
Befindet sich das System beim Start des Programms in einem solch katastrophalen Zustand, daß sich die Bibliotheken nicht einmal mehr öffnen lassen, so erfolgt die Ausgabe der Fehlermeldung nicht über die sonst übliche Methode mittels „printf()", sondern vielmehr durch „Alert()", da die Wahrscheinlichkeit einer erfolgreichen Ausführung hier höher ist. Die Alerts sind natürlich „recoverable" und erzwingen keinen Reset.
Das Programm wurde mit Lattice-C-4.00 entwickelt und verwendet die mitgelieferten Include-Files. Daher mußten dem Quelltext einige kleine Korrekturen beigefügt werden, die aus Unzulänglichkeiten eben dieser Header resultieren. Die Verwendung von Integers mit 16 Bits Breite verkürzt die Länge des Executables übrigens entgegen der landläufigen Meinung lediglich um rund ein Prozent. Sollten Sie Ihren Compiler also noch nicht in- und auswendig kennen, so empfiehlt sich immer die Selektion von langen Integers.

(Ralph Babel)

Programmname: IconAssem
Sprache: C


Auf Leserdiskette



Überprüfen mit Checksum

```
/*                                                        653
** IconAssem.c - everything that's missing in "IconEd"    653
** Copyright (C) 1988 by Ralph Babel, Falkenweg 3,
 D-6204 Taunusstein                                       653
** all right reserved - alle Rechte vorbehalten           653
*/                                                        358
                                                          358
#ifdef AZTEC_C                                            032
                                                          032
/* haha - pack ein! */                                    902
                                                          902
#else                                                     234
                                                          234
/*** included files ***/                                  008
                                                          008
#include <exec/types.h>                                   744
#include <exec/alerts.h>                                  682
#include <exec/execbase.h>                                057
#include <exec/memory.h>                                  817
#include <exec/resident.h> /* missing in <proto/exec.h>
 */                                                       804
#include <libraries/dos.h>                                817
#include <libraries/dosextens.h>                          989
#include <intuition/intuition.h>                          797
#include <intuition/intuitionbase.h>                      640
#include <workbench/startup.h>                            916
#include <workbench/workbench.h>                          540
#include <clib/macros.h>                                  273
                                                          273
/*** lattice special (prototypes, builtin) ***/           775
                                                          775
#include <proto/exec.h>                                   034
#include <proto/dos.h>                                    818
#include <proto/intuition.h>                              159
#include <proto/icon.h>                                   646
#include <string.h>                                       036
                                                          036
typedef char *XPTR; /* really an APTR, patch for bad
 <proto/*.h> */                                           176
                                                          176
struct Resource { struct Library Library; }; /* patch
 for bad <proto/exec.h> */                                880
                                                          880
struct WBObject { char dummy; }; /* patch for bad
 <proto/icon.h>? */                                       143
                                                          143
/*** external symbol references, amiga.lib ***/           869
                                                          869
void printf(char *, );                                    986
void fprintf(BPTR, char *, );                             225
void exit(LONG); /* really neither "void" nor
 "amiga.lib" ... */                                       819
                                                          819
/*** global symbols for linker & library (bss section)
 ***/                                                     609
                                                          609
extern struct ExecBase *SysBase;                          079
                                                          079
struct IntuitionBase *IntuitionBase;                      501
struct Library *IconBase;                                 340
                                                          340
/*** constants ***/                                       530
                                                          530
#define ZERO 0 /* "NULL" for BPTRs */                     982
                                                          982
```

```
#define EOF (-1)                                              797
#define EOS '\0'                                              344
                                                              344
#define MAXARGV 128                                           467
#define MAXARGB 256                                           188
                                                              188
#define MAXFILENAME  30                                       964
#define MAXFILENAME0 (MAXFILENAME + 1)                        021
#define MAXFILEPATH  255                                      494
#define MAXFILEPATH0 (MAXFILEPATH + 1)                        511
                                                              511
#define MAXTOOLTYPES 128                                      945
                                                              945
#define MSG_USAGE     (-1)                                    393
#define MSG_RANGE     (-2)                                    374
#define MSG_MEMORY    (-3)                                    628
#define MSG_NOTOOL    (-4)                                    709
#define MSG_NODRAWER  (-5)                                    854
#define MSG_NOPROJECT (-6)                                    738
                                                              738
#define NOT_A_NUMBER NO_ICON_POSITION /* heh-heh! */          136
                                                              136
/*** some useful macros ***/                                  646
                                                              646
#define ALLOCSTRUCT(name) ((struct name *)AllocMem(sizeof(struct name),\   320
 MEMF_PUBLIC | MEMF_CLEAR))                                   060
#define FREESTRUCT(name) FreeMem((XPTR)name, sizeof(*name))   221
#define ImageDataSize(ig) (RASSIZE((ig)->Width, (ig)->Height) * (ig)->Depth)   263
                                                              263
/*** structures ***/                                          885
                                                              885
struct Global                                                 791
 {                                                            166
 struct DiskObject DiskObject;                                622
 struct DrawerData DrawerData;                                308
 struct Image Image[2];                                       044
 struct Window *Window[2];                                    316
 char *ToolTypes[MAXTOOLTYPES + 1];                           787
 char CurrentDirName[MAXFILEPATH0];                           338
 UWORD ToolCount;                                             577
 BYTE Running;                                                605
 };                                                           529
                                                              529
struct Command                                                317
 {                                                            148
 char *Name;                                                  888
 char *Usage;                                                 580
 LONG (*Function)(struct Global *, BPTR, UWORD, char *[]);    786
 };                                                           398
                                                              398
/*** code section ***/                                        236
                                                              236
static char *AmigaDOSError(LONG);                             739
                                                              739
static char *AmigaDOSError(code)                              370
LONG code;                                                    232
 {                                                            091
 switch(code)                                                 229
  {                                                           052
  case ERROR_NO_FREE_STORE:                                   789
   return "no free store";                                    212
  case ERROR_TASK_TABLE_FULL:                                 655
   return "task table full";                                  188
  case ERROR_LINE_TOO_LONG:                                   499
   return "line too long";                                    316
  case ERROR_FILE_NOT_OBJECT:                                 589
   return "file not object";                                  912
  case ERROR_INVALID_RESIDENT_LIBRARY:                        583
   return "invalid resident library";                         508
  case ERROR_NO_DEFAULT_DIR:                                  930
   return "no default directory";                             920
  case ERROR_OBJECT_IN_USE:                                   600
   return "object in use";                                    022
  case ERROR_OBJECT_EXISTS:                                   379
   return "object exists";                                    641
  case ERROR_DIR_NOT_FOUND:                                   774
   return "directory not found";                              847
  case ERROR_OBJECT_NOT_FOUND:                                960
   return "object not found";                                 587
  case ERROR_BAD_STREAM_NAME:                                 440
   return "bad stream name";                                  339
  case ERROR_OBJECT_TOO_LARGE:                                160
   return "object too large";                                 375
  case ERROR_ACTION_NOT_KNOWN:                                656
   return "action not knowm";                                 042
  case ERROR_INVALID_COMPONENT_NAME:                          926
   return "invalid component name";                           832
  case ERROR_INVALID_LOCK:                                    474
   return "invalid lock";                                     001
  case ERROR_OBJECT_WRONG_TYPE:                               044
   return "object of wrong type";                             346
  case ERROR_DISK_NOT_VALIDATED:                              025
   return "disk not validated";                               562
  case ERROR_DISK_WRITE_PROTECTED:                            059
   return "disk write protected";                             462
  case ERROR_RENAME_ACROSS_DEVICES:                           201
   return "rename across devices";                            466
  case ERROR_DIRECTORY_NOT_EMPTY:                             292
   return "directory not empty";                              624
  case ERROR_TOO_MANY_LEVELS:                                 491
   return "too many levels";                                  748
  case ERROR_DEVICE_NOT_MOUNTED:                              842
   return "device not mounted";                               510
  case ERROR_SEEK_ERROR:                                      250
   return "seek error";                                       423
  case ERROR_COMMENT_TOO_BIG:                                 487
   return "comment too big";                                  681
  case ERROR_DISK_FULL:                                       953
   return "disk full";                                        354
  case ERROR_DELETE_PROTECTED:                                377
   return "delete protected";                                 029
  case ERROR_WRITE_PROTECTED:                                 652
   return "write protected";                                  648
  case ERROR_READ_PROTECTED:                                  704
   return "read protected";                                   329
  case ERROR_NOT_A_DOS_DISK:                                  516
   return "not a dos disk";                                   522
  case ERROR_NO_DISK:                                         168
   return "no disk";                                          135
  case ERROR_NO_MORE_ENTRIES:                                 566
   return "no more entries";                                  003
  default:                                                    854
   return NULL;                                               984
  }                                                           629
 }                                                            110
                                                              110
static LONG stoL(char *);                                     284
                                                              284
static LONG stoL(s)                                           722
char *s;                                                      156
 {                                                            103
 BYTE sign;                                                   931
 LONG l;                                                      070
                                                              070
 while(*s == ' ' || *s == '\t')                               094
  ++s;                                                        698
                                                              698
 sign = *s == '-'? (++s, -1): 1;                              047
                                                              047
 /* won't check for overflow - caveat emptor! */              285
                                                              285
 for(l = 0; '0' <= *s && *s <= '9'; ++s)                      934
  l = l * 10 + *s - '0'; /* hopefully no strange character set ... */   114
                                                              114
 return *s == EOS? l * sign: NOT_A_NUMBER;                    012
 }                                                            281
                                                              281
static char *DupString(char *);                               409
                                                              409
static char *DupString(s)                                     113
char *s;                                                      539
 {                                                            986
 char *t;                                                     689
                                                              689
 if(s != NULL && (t = (char *)AllocMem((ULONG)(strlen(s) + 1), 0)) != NULL)   359
  {                                                           694
  strcpy(t, s);                                               486
                                                              486
  return t;                                                   549
  }                                                           118
                                                              118
 return NULL;                                                 784
 }                                                            077
                                                              077
static void FreeString(char *);                               536
                                                              536
static void FreeString(s)                                     175
char *s;                                                      537
 {                                                            960
 if(s != NULL)                                                328
  {                                                           291
  FreeMem((XPTR)s, (ULONG)(strlen(s) + 1));                   745
  }                                                           618
 }                                                            015
                                                              015
static void CopyImage(struct Image *, struct Image *);        311
                                                              311
static void CopyImage(srcig, dstig)                           563
struct Image *srcig, *dstig;                                  161
 {                                                            168
 ULONG size, oldsize;                                         821
 UWORD *data, *olddata;                                       272
                                                              272
 oldsize = ImageDataSize(dstig);                              146
 olddata = dstig->ImageData;                                  982
                                                              982
 dstig->ImageData = NULL;                                     652
                                                              652
 if(srcig != NULL)                                            353
  {                                                           616
  if((data = (UWORD *)AllocMem(size = ImageDataSize(srcig), MEMF_CHIP)) != NULL)   569
   {                                                          376
   *dstig = *srcig;                                           443
   CopyMem((XPTR)srcig->ImageData,
    (XPTR)(dstig->ImageData = data), size);                   925
   dstig->PlanePick  = 0x3;                                   037
   dstig->PlaneOnOff = 0x0;                                   619
   dstig->NextImage  = NULL;                                  314
   }                                                          111
  }                                                           568
                                                              568
 if((srcig == NULL || data != NULL) && olddata != NULL)       899
  {                                                           570
  FreeMem((XPTR)olddata, oldsize);                            517
  }                                                           702
 }                                                            059
                                                              059
static LONG MakePath(char *, struct FileInfoBlock *, BPTR, WORD);   700
                                                              700
static LONG MakePath(path, fib, lock, level)                  939
char *path;                                                   579
```

Hier könnte Ihre Anzeige stehen.

Rufen Sie einfach an.

Telefon: 089/3 60 86-201

Wir beraten Sie gerne.

Amiga & Zubehör

A 2000 m. Monitor 1084	2695,—
A 2000 m. Monitor 1084 u. XT-Karte	3395,—
A 2000 AT-Karte/XT-Karte	Preis auf Anfrage
Harddisk für A 500/1000/2000	Preis auf Anfrage
Pal-Video-Karte für A 2000	159,—
2MB-Erweiterung für A 2000	895,—
512 KB-RAM für A 500 akkugep. Uhr/abschaltbar	259,—
Golem 2MB-RAM-Box für A 1000	128,—
NEC 3,5'' Zweitlaufwerk Abschalter/Busdurchführung	298,—
Mitsubishi Multisync-Monitor EUM 1471	1345,—
Datamat-Profimat-Textomat	je 89,—
Becker/Text deutsche Textverarbeitung	189,—
Turboprint	89,—
Jet	89,—
Public-Domain Fish-Disk inkl. 2DD Disk	je 4,—

Disketten Tages- u. Staffelpreise erfragen!

Drucker-Sonderpreise

Neu! Star LC10 Colordrucker	725,—
Star LC10 9-Nadeldrucker	595,—
NEC P2200 24-Nadeldrucker	935,—
NEC P6 Colordrucker	1365,—

Wir liefern weitere Hard- und Software zu günstigen Preisen!

COMPUTER-SHOP-RUTH
Mullstr. 6, 2833 Harpstedt, Tel. 0 42 44/18 77

CPS Computertechnik GmbH

Tel.: (05 31)79 80 10

...weil Preis & Leistung stimmen!

AMIGA 500 965,— / PAL-Modulator 55,— / AMIGA 500 + Philips RGB Monitor 8833 + 512 KB Speichererw. mit Uhr 1835,— / AMIGA 500 + Philips RGB Monitor 8833 + EPSON LX 800 + 512 KB Speichererw. 2333,—/*AMIGA 2000 + Monitor 1084 2700,— / *AMIGA 2000 +PC/XT-Karte u. 5¼ Lw + Monitor 1084 3499,—/AMIGA 2000 interne 2 MB Speichererw. (orig. Commodore) 770,— / AMIGA 2000 Harddisk 20 MB, SCSI Controller 1055,— / Disketten nn 2 DD 10 Stck. 20,— / Drucker-Farbbänder NL/NG//ND/NR-10 13,50./EPSON LX 800/LQ 500 10,80.

* Verkauf solange Vorrat reicht. Zwischenverkauf vorbehalten, Lieferung per UPS-Nachnahme. Fordern Sie unseren Gesamtkatalog gegen DM 3,— in Briefmarken an.

CPS Computertechnik GmbH
Marienstraße 16 — 3300 Braunschweig

DI DI DIGITIZE

S/W Röhrenkamera 1/2 Zoll 528,-
C-Objektivanschluß

Color CCD Kamera 2/3 Zoll 2398,-
C-Objektivanschluß

C-Objektive von 3–350 mm
Macro-, Repro- u.a. Adapter

Color-Fotos von Diskette
Dias, Vergrößerungen ab 2,50

OPTIVISION
Heckenerstr. 16 D-5469 Windhagen
Tel.: 02645/4424

CompTec Aktuell

3,5"-Amiga-Laufwerk extern
formsch. Metallgeh., beige Blende, 880 KB, durchgef. Port, Staubschutz, abschaltb. **289.-**

3,5 "-Amiga-Laufw. intern m. Mat. **228.-**

5,25" -Amiga-Laufwerk extern
formsch. Metallgeh., beige Blende, 40/80 Spur, durchgef. Port, abschaltbar **339.-**

INFOS gegen Freiumschlag erhältlich bei:

CompTec Versand
Obermörmtererstraße 1 - nähe Reeser Rheinbrücke
4192 Kalkar 4 (Niedermörmter)
Tel.: 0 28 24 / 38 67

Funkcenter Mitte GmbH

Klosterstr. 130 · 4000 Düsseldorf 1
Tel. 02 11/36 25 22 · Fax 0211/36 01 95

Forms in Flight	169,—
Videoscape 3D	348,—
Page Flipper	98,—
DeLuxe Paint II	228,—
DeLuxe Video 1.2	228,—
DeLuxe Print	148,—
Aegis Audiomaster	148,—
Aegis Diga	178,—
Marauder II	89,—
AC Basic Compiler	348,—
DOS 2 DOS	128,—

450 Public-Domain-Disketten für AMIGA!
Katalogdiskette gegen 5,— Briefmarken oder Schein anfordern.

Mailbox 24 Std. ONLINE, 02 11/36 01 04 8,N,1

Computerservice
Tino Hofstede
An der Windmühle 8
5010 Bergheim 5

Gegen 2 mal 80 Pf in Briefmarken erhalten Sie den

Programmkatalog

für:
- ·Amiga ·C 16/116
- ·C 128 ·Plus/4
- ·C 64 ·VC 20

mit:
- ernsthaften Programmen
- interessanten Spielen aller Art
- neue Software
- Programme, die Sie nur hier bekommen

Beispiele:

Spielepaket (6 Spiele)	C 64, C 16	K/D	DM 9,90
Adressverwaltung	C 64, C 16	K/D	DM 29,90
Textverarbeitung	C 64, C 16	K/D	DM 9,90
Karteikasten	C 64, C 16	K/D	DM 29,90
Videoverwaltung	C 64, C 16	K/D	DM 19,90
Mathe-/Vokabelprogramme	C 64, C 16	K/D	ab DM 19,90
Musik-/Fotoarchive	C 64, C 16	D	ab DM 29,90
Fibu	C 64, C 16	D	ab DM 29,90
Schaufensterwerbung	C 64	D	DM 98,00
KFZ-Programme	C 64	D	ab DM 9,90
Geschäftsprogramme	C 64, C 16	K/D	ab DM 19,90
Datenbanken	Amiga, 128	D	ab DM 99,00

IDS — Systemberatung — Hard- + Software

3,5" LAUFWERK

- Anschlußfertig
- FLOPPYBUS DURCHGEFÜHRT
- Amigafarbenes Metallgehäuse
- Doppelt abgeschirmt
- 100% kompatibel zu allen Programmen
- Abschaltbar, verhindert Abstürze
- Chinon 354, sehr leise

SONDERPREIS

269,— Tel. 0 72 63/59 93

GET IT

Eine der größten Sammlungen Europas von Amiga Public Domain Disks

über **1000** Disketten im Pool

Fish, Faug, Amicus, Panorama, Auge 4000, AIT-Special (Tornados), Taifun, Casa, UKaug, AMIGAzine, AmigaJiuce, Chiron Conceptions, AIT, ACS, BCS, SACC, S.A.F.E., Demos, Slideshows, Entertain, AUSTRIA Kirschbaum (Spiele), Tutorials, DBW-Render 2.0, Virusprotectoren, Ruhr-PD (Canada), R. Wolf, TBAG, Software Digest, Utilities, Demos, Intros, Basic, Ruhr-Serie,... immer topaktuell, s. B. über 75 ACS Disks, immer direkt Lieferung von Fred Fish an uns, usw.

5,— DM pro Disk pl. Porto/ab 10 Stck. Porto/Verp. frei. Nachnahme plus 3,50 DM.

Super PD-Info auf 4 Disketten = > 2800KB = 15 DM
Lauffähig auf allen AMIGAS, berechtigt auch zum Tausch von 4 zu 1, beschreibt alle Programme im Pool.

Neu das PD-Magazin auf Diskette: „GET IT", jeden Monat neu, randvoll für 10 DM inkl. Porto, mit Kursen, Tips, Infos, Helps über PD-Disks und Preisausschreiben. Jeder kann mitwirken.

> GetiT Abonnenten zahlen nur 4 DM pro Disk aus dem POOL <

A I T c/o M. Rönn, Erlenkamp 13, 4650 Gelsenkirchen
NEUE ANSCHRIFT — ACHTUNG — NEUE ANSCHRIFT

AIT

AMIGA WELT INFOMARKT

Achtung!

1. Software darf nur vom Berechtigten vervielfältigt und verbreitet werden. Originalprogramme sind am Originalaufkleber des Datenträgers (Diskette oder Kassette) und ggf. am Copyright-Vermerk zu erkennen. Auch befinden sie sich normalerweise in der Originalverpackung. Wir empfehlen Käufern dringend, darauf zu achten, daß sie tatsächlich Originalprogramme erwerben.
2. Es verstößt gegen das Urheberrecht bzw. das Gesetz gegen den unlauteren Wettbewerb, Raubkopien herzustellen oder zu verbreiten. Verstöße können zivilrechtlich und ggf. auch strafrechtlich geahndet werden. Käufern von Raubkopien steht kein Nutzungsrecht zu. Sie müssen mit Unterlassungsverfahren und ggf. mit sofortiger Beschlagnahme der erworbenen Raubkopien rechnen.
   Wir erwarten von unseren Inserenten und Lesern, daß sie **keine** Raubkopien verbreiten, anbieten oder erwerben.
3. Anzeigen, die vermuten lassen, daß mit Raubkopien gehandelt wird, werden nicht veröffentlicht.

AMIGA WELT KLEINANZEIGEN

Zählt für Sie jeder Millimeter? Dann liegen Sie hier richtig mit Ihrem Angebot. Nutzen Sie diese Rubrik, um Ihre gewerbliche oder private Kleinanzeige zu plazieren. Wir berechnen DM 7,— zzgl ges. MwSt. pro Zeile für gewerbliche Anzeigen (1 Zeile = 33 Anschläge) und DM 5,— pro 5 Zeilen für private Anzeigen.

Und so wird es gemacht! Schicken Sie uns Ihren Text mit der genauen Adresse und nennen Sie uns die entsprechende Rubrik, unter der Ihre Anzeige erscheinen soll. Weitere Informationen geben wir Ihnen aber auch gerne unter Tel. 089/3 60 86-201.

GEWERBLICHE KLEINANZEIGEN

!! Preiswerte Spitzensoftware !! Amiga Public Domain. Alle üblichen Serien (Fish, Tbag...) sowie Grafik und Sound-Sonderserien. Pro Disk (2DD) ab 4,70. Katalog gegen DM 5.— in bar oder Briefm. Flight II-Anleitung in Deutsch + Karten DM 20. Chris Schumacher, Wormser Str. 9, 5000 Köln 1, Tel: 02 21/37 23 91

Gratisliste über Public-Domain für Amiga anfordern bei F. Neuper, 8473 Pfreimd, Postfach 72. Gratis!

★ ★ ★ Public Domain ★ ★ ★ FAUG, TAIFUN, FISH u.a. pro Disk 5,50 DM, ab 30 Disk 4,50 DM: Peter Keim, Vogelsanger Str. 34, 5000 Köln 30, Tel.: 02 21/52 07 65

Ihre IFF-Grafiken auf Dia
— Super Staffelpreise
— Poster bis 50×70
— s/w Ausdrucke bis DIN A2
Diskette ins Kuvert und ab an:
Studio 5
Schwanthalerstr. 14, 8000 München 2
Hotline 0 89/55 59 57

STUDIO 5 Archivdisketten
— Hintergrund Data-Disk 39,90
— Sculpt/Videoscape 49,90
— Sounds in allen Formaten 39,90
(JFF, Sonix, Mimetics)
— Mouse T-Shirt 14,90
Hotline 0 89/55 59 57

ENDLICH
Schulung für AMIGA
Einsteiger + Fortgeschrittene
- Grundkurs - DPaint - Sculpt 3D -
- Animation -
Hotline 0 89/55 59 57

■ ■ ■ ■ Public-Domain ■ ■ ■ ■
★ Gesamte PD (Fish, Panorama usw)
3,20-4,00 DM auf 2DD Disks
4,00-4,80 DM auf 2 DD Markendisks
★ Info auf 2 Disketten: 5,— DM ★
Schramm PD, Philipp-Holl-Str. 18b
6200 Wiesbaden, Tel. 06121/401709

ACHTUNG AMIGA-USER ACHTUNG
Wir führen Software für Ihren Computer. Fordern Sie unsere kostenlose Liste an.
BERLAU-SOFT, Postfach 14 15
2150 Buxtehude.

Laufwerke für Amiga
Externes Laufwerk in Metallgehäuse, durchgeschliffener BUS, Laufwerk um/abschaltbar,
3½″ 1 MB 2×80 Sp. 290 DM
5¼″ 1 MB 2×80/40 Sp. 375 DM
Angelika Huber Elektr.Bedarf, Wörnitzstr. 3, 8850 Donauwörth, Tel. 0906/5567

■ **DISKETTEN** m. Garant. ■
■ 3,5″, 2DD, 135 tpi **DM 2,59** ■
■ 5,25″ ds, dd 48 tpi **DM 0,79** ■
■ 5,25″ dto, 5-Farben-Pack **DM 1,09** ■
■ **NEU: Disketten-Kopier-Service** ■
■ Allgem. Austro-Agentur B. Goller ■
■ Schleißheimer Str. 16, 8057 Eching ■
■ T: 089/3195456, Fax: 089/3195975 ■

PRIVATE KLEINANZEIGEN

Suche MC # 10/1985.

Suche Amiga-Manuals aus uraltem Developers. Package („weiße" Ausgaben, nicht Addison-Wesley oder Bantam).

Freundlicher Usenet-Host Nähe Wiesbaden gesucht.

Suche Robert Pecks „Programmer's Guide to the Amiga" von Sybex.

Suche englische und amerikanische Developer-Zeitschriften „Amiga-Mail" und „Kickstart" (NICHT das gleichnamige deutsche Magazin!). Auch Kopien angenehm.

I think that I shall never see a poem lovely as an AVL-tree. — Joyce Kilmer (updated)

Science-fiction und Fantasy aller Art, deutsch und englisch, gesucht.

Dynamischer Programmierer sucht junge, sympathische SUN-4, Laserprinter kein Hindernis.

Wer gibt Literatur aus den Gebieten Mathematik, Physik und Informatik (deutsch und englisch) in gute Hände ab?

Ralph Babel, Falkenweg 3, D-6204 Taunusstein, (0 61 28) 4 42 48

SUCHE DEUTSCHE TASTATUR für A1000 zum Tausch gegen meine amerik. Tast. (gut erhalten) evtl. Kauf, Alexander Wörner, 0 81 06/48 55, Finkenstraße 8, 8011 Baldham

Verkaufe: Amiga-Software Monopoly-Spiel (ab 1MB) 29,— / Dictionary (Vokabeltrainer) 19,— / Haushaltsbuch (mit Fixbuch.) 19,— / Alle Programme mausgesteuert und mit Anleitung. Porto inkl. T. Riegel, Dorfstr. 52, 8034 Germering

RAM über 512K? Spiele etc. laufen nicht mehr? Disk NoFastResetfestladen -Reset -OK! 5 DM. Peter Pathe, Oberste Homberg 61, 5620 Velbert 15

1 MB ist für div. Spiele zuviel. Die Lösung: Resetfester 512Kbyte-Begrenzer. Disk laden — Reset-fertig. Für A501/A2000. 5 DM + NN o. RPorto. Pathe, Tel. 0 20 53/33 56

Anzeigenschluß für AmigaWelt 4/88 ist der 15. Juli 1988

```
struct FileInfoBlock *fib;                                  173
BPTR lock;                                                  657
WORD level;                                                 984
 {                                                          627
 LONG result;                                               661
 BPTR parent;                                               698
                                                            698
 if((parent = ParentDir(lock)) != ZERO)                     203
  {                                                         714
  result = MakePath(path, fib, parent, (UWORD)(level +
   1));                                                     560
                                                            560
  UnLock(parent);                                           318
                                                            318
  if(result == 0)                                           847
   {                                                        894
   if(Examine(lock, fib) != DOSFALSE)                       471
    {                                                       102
    if(strlen(path) + strlen(fib->fib_FileName) <=
     MAXFILEPATH)                                           249
     {                                                      312
     strcat(path, fib->fib_FileName);                       159
                                                            159
     if(level != 0)                                         549
      {                                                     116
      if(strlen(path) < MAXFILEPATH)                        009
       {                                                    240
       strcat(path, "/");                                   317
       }                                                    150
      else                                                  863
       result = ERROR_TOO_MANY_LEVELS;                      523
      }                                                     780
     }                                                      025
    else                                                    402
     result = ERROR_TOO_MANY_LEVELS;                        202
    }                                                       703
   else                                                     920
    result = IoErr();                                       417
   }                                                        450
  }                                                         879
 else                                                       816
  {                                                         491
  strcpy(path, 1 + (char *)BADDR(((struct DeviceList *)     011
   BADDR(((struct FileLock
    *)BADDR(lock))->fl_Volume))->dl_Name));                 975
  strcat(path, ":");                                        691
                                                            691
  result = 0;                                               618
  }                                                         015
                                                            015
 return result;                                             193
 }                                                          586
                                                            586
/*** commands & support ***/                                848
                                                            848
static void RemoveImage(struct Global *, UBYTE);            888
                                                            888
static void RemoveImage(g, n)                               337
struct Global *g;                                           619
UBYTE n;                                                    265
 {                                                          520
 if(g->Window[n] != NULL)                                   737
  {                                                         512
  CloseWindow(g->Window[n]);                                702
  g->Window[n] = NULL;                                      545
  }                                                         066
                                                            066
 CopyImage(NULL, &g->Image[n]);                             448
 }                                                          853
                                                            853
static LONG SetCurrentDir(struct Global *, BPTR);           311
                                                            311
static LONG SetCurrentDir(g, lock)                          874
struct Global *g;                                           752
BPTR lock;                                                  044
 {                                                          695
 LONG result;                                               385
 struct FileInfoBlock *fib;                                 475
 char path[MAXFILEPATH0];                                   788
                                                            788
 if(lock != 0)                                              563
  {                                                         298
  if((fib = ALLOCSTRUCT(FileInfoBlock)) != NULL)            454
   {                                                        929
   if(Examine(lock, fib) != DOSFALSE)                       602
    {                                                       805
    if(fib->fib_DirEntryType > 0) /* IsDir? */              440
     {                                                      747
     if((result = MakePath(path, fib, lock, 0)) == 0)       245
      {                                                     260
      lock = CurrentDir(lock);                              577
      strcpy(g->CurrentDirName, path);                      244
      }                                                     249
     }                                                      314
    else                                                    043
     result = ERROR_OBJECT_WRONG_TYPE;                      107
    }                                                       516
   else                                                     149
    result = IoErr();                                       210
   FREESTRUCT(fib);                                         810
   }                                                        415
  else                                                      120
   result = ERROR_NO_FREE_STORE;                            454
                                                            454
  UnLock(lock);                                             887
  }                                                         416
 else                                                       009
  if((result = IoErr()) == 0)                               266
   result = ERROR_OBJECT_NOT_FOUND;                         600
                                                            600
 return result;                                             394
 }                                                          151
                                                            151
static LONG cmd_cd(g, stdout, argc, argv)                   865
struct Global *g;                                           315
BPTR stdout;                                                789
UWORD argc;                                                 722
char *argv[];                                               529
 {                                                          856
 switch(argc)                                               848
  {                                                         907
  case 1:                                                   610
   fprintf(stdout, "%s\n", g->CurrentDirName);              449
   return 0;                                                176
  case 2:                                                   288
   return SetCurrentDir(g, Lock(argv[1], ACCESS_READ));     945
  }                                                         122
                                                            122
 return MSG_USAGE;                                          004
 }                                                          177
                                                            177
static LONG cmd_dir(g, stdout, argc, argv)                  619
struct Global *g;                                           341
BPTR stdout;                                                807
UWORD argc;                                                 116
char *argv[];                                               275
 {                                                          650
 BPTR lock;                                                 534
 LONG result;                                               304
 WORD length;                                               053
 struct FileInfoBlock *fib;                                 583
 char buffer[MAXFILENAME0];                                 109
                                                            109
 if(argc <= 2)                                              589
  {                                                         636
  if(argc == 2)                                             409
   {                                                        344
   if((lock = Lock(argv[1], ACCESS_READ)) == ZERO)          045
    {                                                       708
    return IoErr();                                         893
    }                                                       494
   }                                                        403
  else                                                      404
   {                                                        279
   lock = ((struct Process
    *)FindTask(NULL))->pr_CurrentDir;                       350
   }                                                        579
                                                            579
  if((fib = ALLOCSTRUCT(FileInfoBlock)) != NULL)            351
   {                                                        590
   if(Examine(lock, fib) != DOSFALSE)                       295
    {                                                       910
    if(fib->fib_DirEntryType > 0) /* IsDir? */              881
     {                                                      336
     while(ExNext(lock, fib) != DOSFALSE)                   528
      {                                                     843
      if(fib->fib_DirEntryType > 0) /* IsDir? */            438
       {                                                    721
       fprintf(stdout, "%s (dir)\n", fib->fib_FileName);    412
                                                            412
       /* insert your favorite recursion for "OPT A"
        here! */                                            730
       }                                                    423
      else                                                  232
       {                                                    091
       if((length = strlen(fib->fib_FileName) - 5) > 0)     001
        {                                                   136
        if(stricmp(fib->fib_FileName + length, ".info")
         == 0)                                              341
         {                                                  460
         memcpy(buffer, fib->fib_FileName, length);         208
         buffer[length] = EOS;                              335
         fprintf(stdout, "\x9b1m%s (icon)\x9bm\n",
          buffer);                                          286
         }                                                  795
        }                                                   220
       }                                                    937
      }                                                     018
     }                                                      359
    else                                                    192
     {                                                      571
     result = ERROR_OBJECT_WRONG_TYPE;                      827
     goto skip;                                             638
     }                                                      227
    }                                                       028
                                                            028
   if((result = IoErr()) == ERROR_NO_MORE_ENTRIES)          230
    result = 0;                                             017
                                                            017
   skip:                                                    002
                                                            002
   FREESTRUCT(fib);                                         626
   }                                                        071
  else                                                      392
   result = MSG_MEMORY;                                     598
                                                            598
  if(argc == 2)                                             547
   {                                                        090
   UnLock(lock);                                            979
   }                                                        564
  }                                                         313
 else                                                       154
  result = MSG_USAGE;                                       164
                                                            164
 return result;                                             014
 }                                                          307
                                                            307
static LONG cmd_save(g, stdout, argc, argv)                 277
```

```
struct Global *g;                                                   119
BPTR stdout;                                                        937
UWORD argc;                                                         350
char *argv[];                                                       597
 {                                                                  740
 if(argc == 2)                                                      097
  {                                                                 384
  if(g->DiskObject.do_Gadget.GadgetRender != NULL)                  727
   {                                                                382
   g->ToolTypes[g->ToolCount] = NULL; /* end of tool
    types */                                                        105
                                                                    105
   if(PutDiskObject(argv[1], &g->DiskObject) == 0)                  818
    {                                                               517
    return IoErr();                                                 962
    }                                                               343
   }                                                                488
  else                                                              209
   {                                                                792
   fprintf(stdout, "Icon does not have an image!\n");               134
   }                                                                867
  }                                                                 156
 else                                                               197
  return MSG_USAGE;                                                 915
                                                                    915
 return 0;                                                          434
 }                                                                  671
                                                                    671
static void SetIconType(struct Global *, UBYTE);                    293
                                                                    293
static void SetIconType(g, icontype)                                725
struct Global *g;                                                   799
UBYTE icontype;                                                     974
 {                                                                  497
 switch(g->DiskObject.do_Type = icontype)                           942
  {                                                                 081
  case WBDISK:                                                      715
  case WBDRAWER:                                                    867
  case WBGARBAGE:                                                   575
   g->DiskObject.do_DrawerData = &g->DrawerData;                    945
   break;                                                           937
  case WBTOOL:                                                      130
  case WBPROJECT:                                                   660
   g->DiskObject.do_DrawerData = NULL;                              490
   break;                                                           218
  }                                                                 911
 }                                                                  728
                                                                    728
static LONG SetToolWindow(struct Global *, char *);                 570
                                                                    570
static LONG SetToolWindow(g, s)                                     563
struct Global *g;                                                   125
char *s;                                                            935
 {                                                                  038
 char *t = NULL;                                                    721
                                                                    721
 if(g->DiskObject.do_DrawerData == NULL)                            621
  {                                                                 004
  if(s == NULL || (t = DupString(s)) != NULL)                       691
   {                                                                914
   FreeString(g->DiskObject.do_ToolWindow);                         333
   g->DiskObject.do_ToolWindow = t;                                 415
                                                                    415
   return 0;                                                        974
   }                                                                499
  else                                                              036
   return MSG_MEMORY;                                               218
  }                                                                 911
                                                                    911
 return MSG_NOTOOL;                                                 027
 }                                                                  476
                                                                    476
static LONG SetDefaultTool(struct Global *, char *);                891
                                                                    891
static LONG SetDefaultTool(g, s)                                    601
struct Global *g;                                                   467
char *s;                                                            933
 {                                                                  012
 char *t = NULL;                                                    591
                                                                    591
 switch(g->DiskObject.do_Type)                                      980
  {                                                                 575
  case WBDISK:                                                      577
  case WBPROJECT:                                                   823
   if(s == NULL || (t = DupString(s)) != NULL)                      170
    {                                                               285
    FreeString(g->DiskObject.do_DefaultTool);                       069
    g->DiskObject.do_DefaultTool = t;                               940
                                                                    940
    return 0;                                                       919
    }                                                               832
   else                                                             937
    return MSG_MEMORY;                                              743
  }                                                                 592
                                                                    592
 return MSG_NOPROJECT;                                              805
 }                                                                  350
                                                                    350
static LONG cmd_getdata(struct Global *, BPTR, UWORD,
 char *[]);                                                         636
                                                                    636
static LONG cmd_getdata(g, stdout, argc, argv)                      469
struct Global *g;                                                   567
BPTR stdout;                                                        361
UWORD argc;                                                         542
char *argv[];                                                       021
 {                                                                  396
 struct DiskObject *wo;                                             030
 char **ps;                                                         634
                                                                    634
 if(argc == 2)                                                      439
  {                                                                 734
  if((wo = GetDiskObject(argv[1])) != NULL)                         035
   {                                                                578
   g->DiskObject.do_Gadget.Width    =
    wo->do_Gadget.Width;                                            449
   g->DiskObject.do_Gadget.Height   =
    wo->do_Gadget.Height;                                           958
   g->DiskObject.do_CurrentX        = wo->do_CurrentX;              623
   g->DiskObject.do_CurrentY        = wo->do_CurrentY;              538
                                                                    538
   SetIconType(g, wo->do_Type);                                     757
   (void)SetDefaultTool(g, wo->do_DefaultTool);                     316
                                                                    316
   if(g->ToolCount == 0 && wo->do_ToolTypes != NULL)                074
    {                                                               085
    for(ps = wo->do_ToolTypes; *ps != NULL; ++ps)                   168
     {                                                              259
     if((g->ToolTypes[g->ToolCount] = DupString(*ps)) ==
      NULL)                                                         518
      {                                                             713
      break;                                                        025
      }                                                             450
     ++g->ToolCount;                                                488
     }                                                              325
    }                                                               254
                                                                    254
   if(g->DiskObject.do_DrawerData != NULL)                          698
    {                                                               005
    g->DrawerData.dd_NewWindow.LeftEdge =
     wo->do_DrawerData->dd_NewWindow.LeftEdge;                      319
    g->DrawerData.dd_NewWindow.TopEdge  =
     wo->do_DrawerData->dd_NewWindow.TopEdge;                       365
    g->DrawerData.dd_NewWindow.Width    =
     wo->do_DrawerData->dd_NewWindow.Width;                         355
    g->DrawerData.dd_NewWindow.Height   =
     wo->do_DrawerData->dd_NewWindow.Height;                        475
    g->DrawerData.dd_CurrentX           =
     wo->do_DrawerData->dd_CurrentX;                                383
    g->DrawerData.dd_CurrentY           =
     wo->do_DrawerData->dd_CurrentY;                                657
    }                                                               474
   else                                                             715
    {                                                               226
    g->DiskObject.do_StackSize = wo->do_StackSize;                  195
    (void)SetToolWindow(g, wo->do_ToolWindow);                      848
    }                                                               909
                                                                    909
   FreeDiskObject(wo);                                              527
   }                                                                832
  else                                                              937
   return IoErr();                                                  014
  }                                                                 307
 else                                                               820
  return MSG_USAGE;                                                 038
                                                                    038
 return 0;                                                          033
 }                                                                  554
                                                                    554
static void SetHighLight(struct Global *, UWORD);                   157
                                                                    157
static void SetHighLight(g, mode)                                   176
struct Global *g;                                                   766
UWORD mode;                                                         059
 {                                                                  890
 g->DiskObject.do_Gadget.Flags =                                    546
  g->DiskObject.do_Gadget.Flags & ~GADGHIGHBITS | mode;             986
                                                                    986
 if(mode != GADGHIMAGE)                                             375
  {                                                                 902
  g->DiskObject.do_Gadget.SelectRender = NULL;                      611
                                                                    611
  RemoveImage(g, 1);                                                276
  }                                                                 665
 }                                                                  578
                                                                    578
static LONG cmd_getimage(struct Global *, BPTR, UWORD,
 char *[]);                                                         713
                                                                    713
static LONG cmd_getimage(g, stdout, argc, argv)                     591
struct Global *g;                                                   209
BPTR stdout;                                                        995
UWORD argc;                                                         864
char *argv[];                                                       079
 {                                                                  150
 static struct NewWindow newWindowTemplate =                        312
  {                                                                 083
  0, 0, 0, 0,                                                       159
  -1, -1,                                                           503
  0,                                                                707
  WINDOWDRAG | WINDOWDEPTH | SMART_REFRESH |
   NOCAREREFRESH,                                                   982
  NULL,                                                             689
  NULL,                                                             552
  NULL,                                                             635
  NULL,                                                             650
  NULL,                                                             533
  0, 0,                                                             041
  0, 0,                                                             389
  WBENCHSCREEN                                                      032
  };                                                                948
                                                                    948
 LONG result;                                                       390
 WORD temp;                                                         451
 ULONG src, dst;                                                    742
 struct DiskObject *wo;                                             168
 struct Image *srcig, *dstig;                                       454
 struct Window *w;                                                  591
```



```
struct Screen S;                                                  774
struct NewWindow NW;                                              741
                                                                  741
result = MSG_USAGE;                                               035
                                                                  035
if(argc >= 2)                                                     565
 {                                                                324
 if((dst = stoL(argv[1])) >= 0 && dst <= 1)                       053
  {                                                               812
  if(argc == 3 || argc == 4)                                      233
   {                                                              104
   if((src = argc == 4? stoL(argv[3]): 0) >= 0 && src
    <= 1)                                                         049
    {                                                             760
    result = 0;                                                   971
                                                                  971
    if((wo = GetDiskObject(argv[2])) != NULL)                     893
     {                                                            492
     if(src == 0 || (wo->do_Gadget.Flags &
      GADGHIGHBITS) == GADGHIMAGE)                                660
      {                                                           511
      if(GetScreenData((XPTR)&S, sizeof(struct Screen),
       WBENCHSCREEN, NULL))                                       206
       {                                                          753
       srcig = (struct Image *)                                   177
        (src == 0? wo->do_Gadget.GadgetRender:
        wo->do_Gadget.SelectRender);                              814
       dstig = &g->Image[dst];                                    055
                                                                  055
       CopyImage(srcig, dstig);                                   536
                                                                  536
       if(g->Image[dst].ImageData != NULL)                        048
        {                                                         747
        if(dst == 0)                                              212
         {                                                        831
         g->DiskObject.do_Gadget.GadgetRender =
          (APTR)dstig;                                            840
         }                                                        805
        else                                                      838
         {                                                        777
         g->DiskObject.do_Gadget.SelectRender =
          (APTR)dstig;                                            302
         SetHighLight(g, GADGHIMAGE);                             707
         }                                                        124
        }                                                         737
                                                                  737
       NW = newWindowTemplate;                                    496
                                                                  496
       NW.Width = dstig->Width + S.WBorLeft +
        S.WBorRight;                                              114
       NW.LeftEdge = (temp = S.MouseX - NW.Width / 2) <
        0? 0:                                                     034
        temp + NW.Width > S.Width? S.Width - NW.Width:
         temp;                                                    876
                                                                  876
       NW.Height = dstig->Height + S.BarHeight +
        S.BarVBorder + S.WBorBottom;                              901
       NW.TopEdge = (temp = S.MouseY - NW.Height / 2) <
        0? 0:                                                     547
        temp + NW.Height > S.Height? S.Height -
         NW.Height: temp;                                         049
                                                                  049
       if((w = OpenWindow(&NW)) != NULL)                          809
        {                                                         400
        DrawImage(w->RPort, dstig, (LONG)w->BorderLeft,
         (LONG)w->BorderTop);                                     793
                                                                  793
        if(g->Window[dst] != NULL)                                019
         {                                                        370
         CloseWindow(g->Window[dst]);                             805
         }                                                        350
                                                                  350
        g->Window[dst] = w;                                       434
        }                                                         671
       }                                                          656
      else                                                        993
       result = MSG_MEMORY;                                       023
      }                                                           424
     else                                                         121
      fprintf(stdout, "\"%s\" does not have an
       alternate image\n", argv[2]);                              015
                                                                  015
      FreeDiskObject(wo);                                         033
      }                                                           554
     else                                                         027
      result = IoErr();                                           520
     }                                                            741
    }                                                             566
   else                                                           743
    {                                                             590
    result = 0;                                                   665
                                                                  665
    *(dst == 0? &g->DiskObject.do_Gadget.GadgetRender:            009
     &g->DiskObject.do_Gadget.SelectRender) = NULL;               157
                                                                  157
    RemoveImage(g, (UBYTE)dst);                                   014
                                                                  014
    if(dst == 1)                                                  276
     SetHighLight(g, GADGHCOMP);                                  237
    }                                                             158
   }                                                              131
  else                                                            924
   result = MSG_RANGE;                                            918
  }                                                               819
                                                                  819
 return result;                                                   557
 }                                                                222
                                                                  222
static LONG cmd_icontype(g, stdout, argc, argv)                   380
struct Global *g;                                                 114
BPTR stdout;                                                      820
UWORD argc;                                                       549
char *argv[];                                                     512
 {                                                                635
 static char *icontype[] =                                        327
  {                                                               278
  NULL,                                                           753
  "DISK",                                                         364
  "DRAWER",                                                       137
  "TOOL",                                                         031
  "PROJECT",                                                      354
  "GARBAGE",                                                      651
  "DEVICE",                                                       011
  "KICK",                                                         489
  NULL                                                            444
  };                                                              944
                                                                  944
 char **ps;                                                       540
                                                                  540
 switch(argc)                                                     188
  {                                                               519
  case 1:                                                         358
   fprintf(stdout, "IconType = %s\n",
    icontype[g->DiskObject.do_Type]);                             493
   break;                                                         237
  case 2:                                                         025
   for(ps = &icontype[1]; *ps != NULL; ++ps)                      424
    if(stricmp(argv[1], *ps) == 0)                                599
     {                                                            766
     SetIconType(g, (UBYTE)(ps - icontype));                      570
     goto skip;                                                   949
     }                                                            174
   fprintf(stdout, "Unknown icon type \"%s\"\n",
    argv[1]);                                                     780
   skip:                                                          769
   break;                                                         097
  default:                                                        908
   return MSG_USAGE;                                              622
  }                                                               019
                                                                  019
 return 0;                                                        026
 }                                                                463
                                                                  463
static LONG SetLONG(BPTR, UWORD, char *[], char *, char
   *, LONG *, LONG *);                                            772
                                                                  772
static LONG SetLONG(stdout, argc, argv, s1, s2, p1, p2)           141
BPTR stdout;                                                      223
UWORD argc;                                                       932
char *argv[];                                                     923
char *s1, *s2;                                                    201
LONG *p1, *p2;                                                    231
 {                                                                078
 LONG l1, l2;                                                     188
                                                                  188
 switch(argc)                                                     372
  {                                                               863
  case 1:                                                         286
   fprintf(stdout, "%s = %ld, %s = %ld\n", s1, *p1, s2,
    *p2);                                                         540
   break;                                                         252
  case 3:                                                         809
   if((l1 = stoL(argv[1])) != NOT_A_NUMBER                        146
   && (l2 = stoL(argv[2])) != NOT_A_NUMBER)                       583
    {                                                             558
    *p1 = l1;                                                     026
    *p2 = l2;                                                     812
    }                                                             441
   else                                                           282
    return MSG_RANGE;                                             108
   break;                                                         204
  }                                                               729
                                                                  729
 return 0;                                                        512
 }                                                                637
                                                                  637
static LONG SetWORD(BPTR, UWORD, char *[], char *, char
   *, WORD *, WORD *);                                            490
                                                                  490
```

```
static LONG SetWORD(stdout, argc, argv, s1, s2, p1, p2)      483
BPTR stdout;                                                  853
UWORD argc;                                                   018
char *argv[];                                                 593
char *s1, *s2;                                                807
WORD *p1, *p2;                                                517
 {                                                            700
 LONG result;                                                 502
 LONG l1, l2;                                                 460
                                                              460
 l1 = *p1;                                                    416
 l2 = *p2;                                                    878
 result = SetLONG(stdout, argc, argv, s1, s2, &l1, &l2);      602
 *p1 = (UWORD)l1;                                             896
 *p2 = (UWORD)l2;                                             072
                                                              072
 return result;                                               506
 }                                                            559
                                                              559
static LONG cmd_iconsize(g, stdout, argc, argv)               874
struct Global *g;                                             752
BPTR stdout;                                                  594
UWORD argc;                                                   371
char *argv[];                                                 022
 {                                                            409
 UWORD w0, h0, w1, h1;                                        393
                                                              393
 switch(argc)                                                 025
  {                                                           448
  case 1:                                                     075
  case 3:                                                     572
   return SetWORD(stdout, argc, argv, "Icon: Width",
    "Height",                                                 400
    &g->DiskObject.do_Gadget.Width,
     &g->DiskObject.do_Gadget.Height);                        157
  case 2:                                                     137
   if(stricmp(argv[1], "adjust") == 0)                        378
    {                                                         941
    w0 = h0 = w1 = h1 = 0;                                    432
                                                              432
    if(g->DiskObject.do_Gadget.GadgetRender != NULL)          799
     {                                                        270
     w0 = g->Image[0].Width;                                  308
     h0 = g->Image[0].Height;                                 600
     }                                                        781
                                                              781
    if(g->DiskObject.do_Gadget.SelectRender != NULL)          904
     {                                                        635
     w1 = g->Image[1].Width;                                  539
     h1 = g->Image[1].Height;                                 133
     }                                                        854
                                                              854
    g->DiskObject.do_Gadget.Width  = MAX(w0, w1);             202
    g->DiskObject.do_Gadget.Height = MAX(h0, h1);             005
                                                              005
    return 0;                                                 164
    }                                                         209
   break;                                                     945
  }                                                           122
                                                              122
 return MSG_USAGE;                                            004
 }                                                            177
                                                              177
static LONG cmd_iconposition(g, stdout, argc, argv)           978
struct Global *g;                                             568
BPTR stdout;                                                  794
UWORD argc;                                                   731
char *argv[];                                                 990
 {                                                            705
 switch(argc)                                                 001
  {                                                           136
  case 1:                                                     331
   if(g->DiskObject.do_CurrentX == NO_ICON_POSITION           256
   && g->DiskObject.do_CurrentX == NO_ICON_POSITION)          935
    {                                                         038
    fprintf(stdout, "Icon: NO POSITION\n");                   957
                                                              957
    return 0;                                                 020
    }                                                         385
  case 3:                                                     674
   return SetLONG(stdout, argc, argv, "Icon: XPosition",
    "YPosition",                                              624
    &g->DiskObject.do_CurrentX,
     &g->DiskObject.do_CurrentY);                             343
  case 2:                                                     027
   if(stricmp(argv[1], "none") == 0)                          785
    {                                                         088
    g->DiskObject.do_CurrentX = NO_ICON_POSITION;             592
    g->DiskObject.do_CurrentY = NO_ICON_POSITION;             057
                                                              057
    return 0;                                                 992
    }                                                         733
   break;                                                     629
  }                                                           110
                                                              110
 return MSG_USAGE;                                            104
 }                                                            477
                                                              477
static LONG cmd_highlight(g, stdout, argc, argv)              904
struct Global *g;                                             526
BPTR stdout;                                                  040
UWORD argc;                                                   193
char *argv[];                                                 580
 {                                                            519
 static char *highlight[] =                                   028
  {                                                           487
  "COMP",                                                     362
  "BACKFILL",                                                 598
  "IMAGE",                                                    489
  "NONE",                                                     901
  NULL                                                        768
  };                                                          404
                                                              404
 WORD mode;                                                   260
                                                              260
 switch(argc)                                                 828
  {                                                           647
  case 1:                                                     998
   fprintf(stdout, "HighLight = %s\n",                        573
    highlight[g->DiskObject.do_Gadget.Flags &
     GADGHIGHBITS]);                                          159
   break;                                                     911
  case 2:                                                     003
   if((mode = stricmp(argv[1], "comp") == 0? GADGHCOMP:       248
    stricmp(argv[1], "backfill") == 0? GADGBACKFILL: -1)
     != -1)                                                   674
    {                                                         693
    SetHighLight(g, mode);                                    969
    }                                                         434
   else                                                       059
    fprintf(stdout, "Unknown highlight type \"%s\"\n",
     argv[1]);                                                688
   break;                                                     008
  default:                                                    019
   return MSG_USAGE;                                          009
  }                                                           242
                                                              242
 return 0;                                                    485
 }                                                            286
                                                              286
static LONG cmd_windowsize(g, stdout, argc, argv)             272
struct Global *g;                                             014
BPTR stdout;                                                  528
UWORD argc;                                                   433
char *argv[];                                                 860
 {                                                            063
 if(g->DiskObject.do_DrawerData != NULL)                      223
  {                                                           974
  switch(argc)                                                742
   {                                                          577
   case 1:                                                    560
   case 3:                                                    173
    return SetWORD(stdout, argc, argv, "Window: Width",
     "Height",                                                032
     &g->DrawerData.dd_NewWindow.Width,
      &g->DrawerData.dd_NewWindow.Height);                    813
   default:                                                   960
    return MSG_USAGE;                                         554
   }                                                          183
  }                                                           456
                                                              456
 return MSG_NODRAWER;                                         171
 }                                                            300
                                                              300
static LONG cmd_windowposition(g, stdout, argc, argv)         964
struct Global *g;                                             026
BPTR stdout;                                                  580
UWORD argc;                                                   117
char *argv[];                                                 808
 {                                                            387
 if(g->DiskObject.do_DrawerData != NULL)                      667
  {                                                           602
  switch(argc)                                                410
   {                                                          357
   case 1:                                                    892
   case 3:                                                    913
    return SetWORD(stdout, argc, argv, "Window:
     XPosition", "YPosition",                                 942
     &g->DrawerData.dd_NewWindow.LeftEdge,
      &g->DrawerData.dd_NewWindow.TopEdge);                   246
   default:                                                   681
    return MSG_USAGE;                                         295
   }                                                          912
  }                                                           741
                                                              741
 return MSG_NODRAWER;                                         112
 }                                                            581
                                                              581
static LONG cmd_windoworigin(g, stdout, argc, argv)           524
struct Global *g;                                             962
BPTR stdout;                                                  364
UWORD argc;                                                   957
char *argv[];                                                 336
 {                                                            395
 if(g->DiskObject.do_DrawerData != NULL)                      451
  {                                                           890
  switch(argc)                                                186
   {                                                          493
   case 1:                                                    292
   case 3:                                                    753
    return SetLONG(stdout, argc, argv, "Window:
     XOrigin", "YOrigin",                                     576
     &g->DrawerData.dd_CurrentX,
      &g->DrawerData.dd_CurrentY);                            227
   default:                                                   054
    return MSG_USAGE;                                         936
   }                                                          005
  }                                                           190
                                                              190
 return MSG_NODRAWER;                                         385
 }                                                            034
                                                              034
static LONG cmd_tooltype(g, stdout, argc, argv)               701
struct Global *g;                                             999
BPTR stdout;                                                  001
UWORD argc;                                                   942
char *argv[];                                                 869
 {                                                            180
```

```
WORD i; /* signed for comparison! */                              034
char *s;                                                          980
LONG l;                                                           667
                                                                  667
if(argc == 1)                                                     815
 {                                                                478
 if(g->ToolCount != 0)                                            061
  for(i = 0; i < g->ToolCount; ++i)                               446
   fprintf(stdout, "%d = \"%s\"\n", i,
    g->ToolTypes[i]);                                             025
 else                                                             402
  fprintf(stdout, "No Tool Types defined.\n");                    025
 }                                                                450
else if(argc == 3 && stricmp(argv[2], "del") == 0)                077
 {                                                                124
 if((l = stoL(argv[1])) >= 0 && l < g->ToolCount)                 990
  {                                                               705
  FreeString(g->ToolTypes[l]);                                    095
  if(--g->ToolCount != 0)                                         808
   for(i = l; i < g->ToolCount; ++i)                              513
    g->ToolTypes[i] = g->ToolTypes[i + 1];                        181
  else                                                            470
   g->DiskObject.do_ToolTypes = NULL;                             380
  }                                                               969
 else                                                             066
  return MSG_RANGE;                                               908
 }                                                                689
else if(argc == 2 || argc == 3)                                   849
 {                                                                920
 if(g->ToolCount < MAXTOOLTYPES)                                  830
  {                                                               673
  if(argc == 3)                                                   847
   {                                                              894
   if((l = stoL(argv[2])) < 0 || l >= g->ToolCount)               285
    return MSG_RANGE;                                             619
   }                                                              028
  else                                                            069
   l = g->ToolCount;                                              102
                                                                  102
  if((s = DupString(argv[1])) != NULL)                            226
   {                                                              013
   if(g->ToolCount != 0)                                          552
    for(i = g->ToolCount - 1; i >= l; --i)                        534
     g->ToolTypes[i + 1] = g->ToolTypes[i];                       402
   else                                                           515
    g->DiskObject.do_ToolTypes = g->ToolTypes;                    963
                                                                  963
   ++g->ToolCount;                                                029
                                                                  029
   g->ToolTypes[l] = s;                                           025
   }                                                              450
  else                                                            331
   return MSG_MEMORY;                                             881
  }                                                               338
 else                                                             475
  fprintf(stdout, "Table of Tool Types full!\n");                 925
 }                                                                910
else                                                              471
 return MSG_USAGE;                                                021
                                                                  021
return 0;                                                         884
}                                                                 377
                                                                  377
static LONG cmd_defaulttool(g, stdout, argc, argv)                031
struct Global *g;                                                 097
BPTR stdout;                                                      603
UWORD argc;                                                       568
char *argv[];                                                     967
 {                                                                406
 switch(g->DiskObject.do_Type)                                    679
  {                                                               758
  case WBDISK:                                                    992
  case WBPROJECT:                                                 874
   switch(argc)                                                   450
    {                                                             877
    case 1:                                                       164
     if(g->DiskObject.do_DefaultTool != NULL)                     764
      fprintf(stdout, "Default Tool = \"%s\"\n",
       g->DiskObject.do_DefaultTool);                             157
     else                                                         086
      fprintf(stdout, "No Default Tool defined.\n");              001
     return 0;                                                    496
    case 2:                                                       840
     return SetDefaultTool(g, stricmp(argv[1], "none")
      == 0? NULL: argv[1]);                                       377
    default:                                                      908
     return MSG_USAGE;                                            622
    }                                                             019
   break;                                                         635
  }                                                               188
                                                                  188
 return MSG_NOPROJECT;                                            889
 }                                                                442
                                                                  442
static LONG cmd_toolwindow(g, stdout, argc, argv)                 451
struct Global *g;                                                 693
BPTR stdout;                                                      623
UWORD argc;                                                       604
char *argv[];                                                     859
 {                                                                050
 if(g->DiskObject.do_DrawerData == NULL)                          818
  {                                                               517
  switch(argc)                                                    469
   {                                                              076
   case 1:                                                        391
    if(g->DiskObject.do_ToolWindow != NULL)                       774
     fprintf(stdout, "Tool Window = \"%s\"\n",
      g->DiskObject.do_ToolWindow);                               053
    else                                                          342
     fprintf(stdout, "No Tool Window defined\n");                 206
    break;                                                        926
   case 2:                                                        774
    return SetToolWindow(g, stricmp(argv[1], "none") ==
     0? NULL: argv[1]);                                           922
   default:                                                       461
    return MSG_USAGE;                                             763
   }                                                              852
  }                                                               961
 else                                                             954
  return MSG_NOTOOL;                                              270
                                                                  270
 return 0;                                                        209
 }                                                                794
                                                                  794
static LONG cmd_stacksize(g, stdout, argc, argv)                  850
struct Global *g;                                                 952
BPTR stdout;                                                      130
UWORD argc;                                                       355
char *argv[];                                                     070
 {                                                                033
 ULONG size;                                                      504
                                                                  504
 if(g->DiskObject.do_DrawerData == NULL)                          872
  {                                                               219
  switch(argc)                                                    459
   {                                                              994
   case 1:                                                        461
    fprintf(stdout, "Stack Size = %ld\n",
     g->DiskObject.do_StackSize);                                 447
    break;                                                        943
   case 2:                                                        187
    if((size = stoL(argv[1])) != NOT_A_NUMBER)                    303
     {                                                            014
     g->DiskObject.do_StackSize = size;                           434
     }                                                            671
    break;                                                        423
   default:                                                       378
    return MSG_USAGE;                                             284
   }                                                              769
  }                                                               930
 else                                                             299
  return MSG_NOTOOL;                                              743
                                                                  743
 return 0;                                                        374
 }                                                                891
                                                                  891
static LONG cmd_quit(g, stdout, argc, argv)                       325
struct Global *g;                                                 719
BPTR stdout;                                                      641
UWORD argc;                                                       046
char *argv[];                                                     509
{                                                                 596
if(argc == 1)                                                     516
 g->Running = FALSE;                                              006
                                                                  006
return argc == 1? 0: MSG_USAGE;                                   409
}                                                                 346
                                                                  346
static LONG cmd_help();                                           566
                                                                  566
static struct Command Table[] =                                   267
 {                                                                546
 "CD",               "[<DirName>]",
                               cmd_cd,                            209
 "Dir",              "[<DirName>]",
                               cmd_dir,                           416
 "Save",             "<IconName>",
                                cmd_save,                         017
 "GetData",          "<IconName>",
                                cmd_getdata,                      804
 "GetImage",         "<DstImage> [<IconName>
  [<SrcImage>]]",   cmd_getimage,                                 042
 "IconType",         "[DISK|DRAWER|TOOL|PROJECT|GARBAGE]",
         cmd_icontype,                                            525
 "IconSize",         "[(<Width> <Height>) | ADJUST]",
              cmd_iconsize,                                       309
 "IconPosition",     "[(<XPosition> <YPosition>) | NONE]",
         cmd_iconposition,                                        676
 "HighLight",        "[COMP|BACKFILL]",
                           cmd_highlight,                         330
 "WindowSize",       "[<Width> <Height>]",
                      cmd_windowsize,                             900
 "WindowPosition",   "[<XPosition> <YPosition>]",
                cmd_windowposition,                               716
 "WindowOrigin",     "[<XOrigin> <YOrigin>]",
                     cmd_windoworigin,                            608
 "ToolType",         "[(<String> [<Tool#>]) | (<Tool#>
  DEL)]", cmd_tooltype,                                           629
 "DefaultTool",      "[<DefaultTool> | NONE]",
                    cmd_defaulttool,                              267
 "ToolWindow",       "[<ToolWindow> | NONE]",
                     cmd_toolwindow,                              214
 "StackSize",        "[<StackSize>]",
                                 cmd_stacksize,                   862
 "Quit",             "",
                                             cmd_quit,            959
 "Help",             "",
                                             cmd_help,            600
 NULL                                                             411
 };                                                               511
                                                                  511
static LONG cmd_help(g, stdout, argc, argv)                       007
struct Global *g;                                                 297
BPTR stdout;                                                      187
UWORD argc;                                                       048
char *argv[];                                                     479
 {                                                                206
 struct Command *pc;                                              630
```

```
                                                                    630
 if(argc == 1)                                                      438
  for(pc = Table; pc->Name != NULL; ++pc)                           617
   fprintf(stdout, "%-14s %s\n", pc->Name, pc->Usage);              175
                                                                    175
 return argc == 1? 0: MSG_USAGE;                                    038
 }                                                                  619
                                                                    619
/*** main loop & local support ***/                                 281
                                                                    281
static LONG fgetc(BPTR);                                            470
                                                                    470
static LONG fgetc(fh)                                               750
BPTR fh;                                                            247
 {                                                                  286
 UBYTE c;                                                           421
                                                                    421
 return Read(fh, &c, 1) == 1? c: EOF;                               530
 }                                                                  871
                                                                    871
static void rditem(char **, char **, UWORD);                        671
                                                                    671
static void rditem(ps, pt, n)                                       228
char **ps; /* source */                                             278
char **pt; /* destination */                                        201
UWORD n;    /* must be >= 1 */                                      205
 {                                                                  740
 char *s, *t;                                                       072
 BYTE quoted;                                                       549
 char delim;                                                        217
                                                                    217
 s = *ps;                                                           053
 t = *pt;                                                           643
                                                                    643
 while(*s == ' ' || *s == '\t')                                     083
  ++s;                                                              919
                                                                    919
 if(quoted = ((delim = *s) == '"' || delim == '\''))                429
  ++s;                                                              377
                                                                    377
 for(; --n != 0 && *s != EOS; )                                     371
  {                                                                 850
  if(quoted? *s == delim && (*(s + 1) == ' ' || *(s + 1)
  == EOS): *s == ' ')                                               866
   {                                                                141
   ++s;                                                             625
   break;                                                           081
   }                                                                178
  *t++ = *s++;                                                      589
  }                                                                 638
                                                                    638
/*                                                                  979
** this means: break "for"-loop if:                                 979
** char == end-of-string OR                                         979
** == QUOTED: char == DELIM && (next-char == SPACE or
EOS)                                                                979
** != QUOTED: char == SPACE                                         979
*/                                                                  156
                                                                    156
 *t++ = EOS;                                                        879
                                                                    879
 *ps = s;                                                           363
 *pt = t;                                                           337
 }                                                                  410
                                                                    410
static LONG IconAssem(char *, char *);                              832
                                                                    832
static LONG IconAssem(console, iconname)                            964
char *console;                                                      294
char *iconname;                                                     655
 {                                                                  446
 struct Global *g;                                                  588
 LONG result, c, code;                                              044
 struct Command *pc;                                                428
 BPTR lock, oldlock;                                                796
 BPTR stdio;                                                        010
 UWORD argc;                                                        139
 char *argv[MAXARGV];                                               096
 char *s, *t, argb[MAXARGB];                                        782
 UBYTE i;                                                           483
                                                                    483
 result = RETURN_ERROR;                                             867
                                                                    867
 if((g = ALLOCSTRUCT(Global)) != NULL)                              431
  {                                                                 630
  if((lock = DupLock(oldlock = CurrentDir(ZERO))) !=
  ZERO)                                                             570
   {                                                                389
   if(SetCurrentDir(g, lock) == 0)                                  254
    {                                                               377
    if((stdio = Open(console, MODE_NEWFILE)) != ZERO)               696
     {                                                              979
     result = RETURN_OK;                                            403
                                                                    403
     g->DiskObject.do_Magic       = WB_DISKMAGIC;                   504
     g->DiskObject.do_Version     = WB_DISKVERSION;                 147
                                                                    147
     /* don't care = ignored by system or IconAssem */              425
                                                                    425
#if 0                                                               915
     g->DiskObject.do_Gadget.NextGadget      = NULL; /*
      don't care */                                                 945
     g->DiskObject.do_Gadget.LeftEdge        = 0;    /*
      don't care */                                                 077
     g->DiskObject.do_Gadget.TopEdge         = 0;    /*
      don't care */                                                 553
     g->DiskObject.do_Gadget.Width           = 0;                   083
     g->DiskObject.do_Gadget.Height          = 0;                   990
#endif                                                              567
                                                                    567
     g->DiskObject.do_Gadget.Flags          = GADGIMAGE |
      GADGHCOMP;                                                    662
     g->DiskObject.do_Gadget.Activation     =
      GADGIMMEDIATE | RELVERIFY;                                    364
     g->DiskObject.do_Gadget.GadgetType     = BOOLGADGET;           048
                                                                    048
#if 0                                                               158
     g->DiskObject.do_Gadget.GadgetRender   = NULL;                 127
     g->DiskObject.do_Gadget.SelectRender   = NULL;                 012
     g->DiskObject.do_Gadget.GadgetText     = NULL; /*
      don't care */                                                 044
     g->DiskObject.do_Gadget.MutualExclude = 0;     /*
      don't care */                                                 938
     g->DiskObject.do_Gadget.SpecialInfo    = NULL; /*
      don't care */                                                 762
     g->DiskObject.do_Gadget.GadgetID       = 0;    /*
      don't care */                                                 027
     g->DiskObject.do_Gadget.UserData       = NULL; /*
      don't care */                                                 343
#endif                                                              144
                                                                    144
     g->DiskObject.do_Type          = WBTOOL;                       829
                                                                    829
#if 0                                                               575
     g->DiskObject.do_DefaultTool = NULL;                           661
     g->DiskObject.do_ToolTypes   = NULL;                           787
#endif                                                              364
                                                                    364
     g->DiskObject.do_CurrentX    =                                 548
     g->DiskObject.do_CurrentY    = NO_ICON_POSITION;               701
                                                                    701
#if 0                                                               959
     g->DiskObject.do_DrawerData  = NULL;                           881
     g->DiskObject.do_ToolWindow  = NULL;                           638
     g->DiskObject.do_StackSize   = 0;                              567
#endif                                                              624
                                                                    624
     g->DrawerData.dd_NewWindow.LeftEdge = 160;                     581
     g->DrawerData.dd_NewWindow.TopEdge  = 50;                      188
     g->DrawerData.dd_NewWindow.Width    = 320;                     299
     g->DrawerData.dd_NewWindow.Height   = 100;                     019
                                                                    019
#if 1                                                               046
     g->DrawerData.dd_NewWindow.DetailPen    = -1;    /*
      don't care */                                                 446
     g->DrawerData.dd_NewWindow.BlockPen     = -1;    /*
      don't care */                                                 510
     g->DrawerData.dd_NewWindow.IDCMPFlags  = 0;      /*
      don't care */                                                 340
     g->DrawerData.dd_NewWindow.Flags        =        /*
      don't care */                                                 702
      WINDOWSIZING | WINDOWDRAG                                     212
      | WINDOWDEPTH | WINDOWCLOSE                                   255
      | SIZEBRIGHT | SIZEBBOTTOM                                    648
      | SIMPLE_REFRESH | REPORTMOUSE                                985
      | WBENCHWINDOW;                                               451
     g->DrawerData.dd_NewWindow.FirstGadget = NULL; /*
      don't care */                                                 922
     g->DrawerData.dd_NewWindow.CheckMark    = NULL; /*
      don't care */                                                 326
     g->DrawerData.dd_NewWindow.Title        = NULL; /*
      don't care */                                                 299
     g->DrawerData.dd_NewWindow.Screen       = NULL; /*
      don't care */                                                 594
     g->DrawerData.dd_NewWindow.BitMap       = NULL; /*
      don't care */                                                 470
     g->DrawerData.dd_NewWindow.MinWidth     = 90;   /*
      don't care */                                                 043
     g->DrawerData.dd_NewWindow.MinHeight    = 40;   /*
      don't care */                                                 416
     g->DrawerData.dd_NewWindow.MaxWidth     = ~0;   /*
      don't care */                                                 984
     g->DrawerData.dd_NewWindow.MaxHeight    = ~0;   /*
      don't care */                                                 389
     g->DrawerData.dd_NewWindow.Type         =
      WBENCHSCREEN; /* don't care */                                326
#endif                                                              607
                                                                    607
#if 0                                                               433
     g->DrawerData.dd_CurrentX = 0;                                 550
     g->DrawerData.dd_CurrentY = 0;                                 984
                                                                    984
     g->Image[0].ImageData =                                        365
     g->Image[1].ImageData = NULL;                                  677
                                                                    677
     g->Window[0] =                                                 688
     g->Window[1] = NULL;                                           610
                                                                    610
     g->ToolCount = 0;                                              383
#endif                                                              400
                                                                    400
     if(iconname != NULL)                                           219
      {                                                             922
      argv[1] = argv[2] = iconname;                                 364
      if(cmd_getdata(g, stdio, 2, argv) == 0)                       384
       {                                                            019
       argv[1] = argv[3] = "0";                                     536
       if(cmd_getimage(g, stdio, 4, argv) == 0)                     611
        {                                                           922
        argv[1] = argv[3] = "1";                                    256
        (void)cmd_getimage(g, stdio, 4, argv);                      111
        }                                                           568
       }                                                            365
      }                                                             774
                                                                    774
     for(g->Running = TRUE; g->Running; )                           031
      {                                                             526
      fprintf(stdio, ".");                                          032
```

```
                                                                 032
        for(s = argb; (c = fgetc(stdio)) != EOF && c !=
          '\n'; )                                                841
         if(s < argb + MAXARGB)                                  549
          *s++ = c;                                              663
                                                                 663
        *s = EOS;                                                255
                                                                 255
        s = argb;                                                530
        t = argb;                                                754
                                                                 754
        for(argc = 0; argv[argc] = t, *s != EOS; ++argc)         057
         {                                                       864
         rditem(&s, &t, (UWORD)(argb + MAXARGB - t));            989
         }                                                       694
                                                                 694
        if(argc != 0)                                            245
         {                                                       260
         for(pc = Table; pc->Name != NULL; ++pc)                 303
          {                                                      014
          if(stricmp(pc->Name, argv[0]) == 0)                    258
           {                                                     429
           switch(code = (*(pc->Function))(g, stdio, argc,
            argv))                                               732
            {                                                    447
            case MSG_USAGE:                                      200
             fprintf(stdio, "Usage: %s %s\n", pc->Name,
              pc->Usage);                                        267
             break;                                              411
            case MSG_RANGE:                                      092
             fprintf(stdio, "?illegal quantity
              error\nready.\n");                                 017
             break;                                              241
            case MSG_MEMORY:                                     210
             fprintf(stdio, "%s: Insufficient free
              store!\n", pc->Name);                              212
             break;                                              012
            case MSG_NOTOOL:                                     859
             fprintf(stdio, "Icon must be of type TOOL or
              PROJECT!\n");                                      796
             break;                                              508
            case MSG_NODRAWER:                                   578
             fprintf(stdio, "Icon must be of type DISK,
              DRAWER or GARBAGE!\n");                            375
             break;                                              959
            case MSG_NOPROJECT:                                  379
             fprintf(stdio, "Icon must be of type DISK or
              PROJECT!\n");                                      669
             break;                                              029
            case 0:                                              519
             break;                                              975
            default:                                             162
             fprintf(stdio, "%s: AmigaDOS error %ld
              (%s)\n", pc->Name, code,                           021
              (s = AmigaDOSError(code)) != NULL? s:
              "unknown");                                        594
             break;                                              026
            }                                                    463
           goto skip;                                            178
           }                                                     391
          }                                                      112
         fprintf(stdio, "Unknown command \"%s\"\n",
          argv[0]);                                              881
                                                                 881
/*                                                               038
** Der Bundesgesundheitsminister: Babel gefährdet Ihren          038
** Compiler. Der Quellcode eines Programms dieses Autors         038
** enthält 0.9 mg Stack Overflow und 13 mg Irreducible           038
** Flow Graphs (GOTO) (Durchschnittswerte nach DIN)              038
*/                                                               871
                                                                 871
         skip:                                                   528
         }                                                       845
        }                                                        870
                                                                 870
      RemoveImage(g, 0);                                         526
      RemoveImage(g, 1);                                         415
                                                                 415
      for(i = g->ToolCount; i != 0; )                            551
       FreeString(g->ToolTypes[--i]);                            452
                                                                 452
      FreeString(g->DiskObject.do_DefaultTool);                  392
      FreeString(g->DiskObject.do_ToolWindow);                   259
                                                                 259
      Close(stdio);                                              768
      }                                                          917
     }                                                           806
    UnLock(CurrentDir(oldlock));                                 749
    }                                                            670
  FREESTRUCT(g);                                                 912
  }                                                              741
                                                                 741
 return result;                                                  527
 }                                                               832
                                                                 832
/*** entry point ***/                                            166
                                                                 166
union argv                                                       207
 {                                                               766
 char **cliArg;                                                  549
 struct WBStartup *wbArg;                                        246
 };                                                              250
                                                                 250
void main(argc, argv)                                            187
ULONG argc;                                                      064
union argv argv;                                                 660
 {                                                               511
 static char window[] = "CON:160/50/320/100/IconAssem";          831
                                                                 831
LONG result;                                                     929
BPTR oldlock;                                                    732
struct WBArg *wa;                                                645
                                                                 645
result = RETURN_FAIL; /* no libraries */                         353
                                                                 353
if((IntuitionBase = (struct IntuitionBase *)                     373
 OpenLibrary("intuition.library", LIBRARY_VERSION)) !=
  NULL)                                                          508
 {                                                               583
 if((IconBase = OpenLibrary("icon.library",
  LIBRARY_VERSION)) != NULL)                                     192
  {                                                              571
  result = RETURN_WARN; /* \n?syntax error\nready.\n */          835
                                                                 835
  if(argc != 0) /* CLI */                                        244
   {                                                             247
   if(argc == 1 || argc == 2 && strcmp(argv.cliArg[1],
    "?") != 0)                                                   904
    {                                                            635
    /* don't request input from background CLI! */               553
                                                                 553
    result = IconAssem(((struct CommandLineInterface *)          071
     BADDR(((struct Process
      *)FindTask(NULL))->pr_CLI))->cli_Background ==             970
     DOSFALSE? "*": window, argc == 2? argv.cliArg[1]:
      NULL);                                                     300
    }                                                            977
   else                                                          178
    printf("Usage from CLI: [RUN] %s [<IconName>]\n",
     argv.cliArg[0]);                                            071
   }                                                             048
  else /* WorkBench */                                           867
   {                                                             154
   switch(argv.wbArg->sm_NumArgs)                                377
    {                                                            928
    case 1:                                                      451
     result = IconAssem(window, NULL);                           348
     break;                                                      548
    case 2:                                                      972
     wa = &argv.wbArg->sm_ArgList[1];                            816
     oldlock = CurrentDir(wa->wa_Lock);                          362
     result = IconAssem(window, wa->wa_Name);                    707
     (void)CurrentDir(oldlock);                                  908
     break;                                                      748
    default:                                                     863
     DisplayBeep(NULL); /* "[...] is not a polite thing
      to do [...]" */                                            902
     break;                                                      662
    }                                                            539
   }                                                             988
  CloseLibrary(IconBase);                                        243
  }                                                              236
 else                                                            509
  Alert(AG_OpenLib | 0xffffL & AO_IconLib,
   (XPTR)&SysBase->ThisTask);                                    679
 CloseLibrary((struct Library *)IntuitionBase);                  497
 }                                                               442
else                                                             171
 Alert(AG_OpenLib | 0xffffL & AO_Intuition,
  (XPTR)&SysBase->ThisTask);                                     488
                                                                 488
exit(result);                                                    437
}                                                                710
                                                                 710
#endif                                                           967
```

Der komplette C-Source um Icons zu bearbeiten

Über 34K Listing sind garantiert nicht als Aufforderung zum Abtippen gedacht. Vielmehr dient der Ausdruck dokumentarischen Zwecken. Die immerhin noch 13K lange Hexdump-Liste haben wir uns und den Lesern aus Platzgründen erspart. Daher die Bitte an alle, die mit IconAssem arbeiten wollen: Nutzen Sie den Leserservice um fertiges Programm und Source auf Diskette zu erhalten.

```
;                                                                059
;IconAssem.bat                                                   513
;                                                                584
FailAt 1                                                         650
LC1 -b0 -ceft -oT: -w IconAssem                                  036
LC2 -r1 -s -v T:IconAssem                                        504
BLink LIB:Astartup.obj+T:IconAssem.o TO IconAssem LIB
  LIB:amiga.lib+LIB:lcsnb.lib SC SD ND                           069
Delete T:IconAssem.o                                             249
FailAt 10                                                        999
```

Listing 2: Batchfile zu IconAssem.

Breakout in neuem Glanz

Die Automatenversion von „Arkanoid" faszinierte in den Spielhallen. Jetzt verfallen Amiga-Besitzer dem fesselnden Breakout. Byte für Byte wurde die klassische Vorlage umgesetzt.

Anfang 1987 wurde ein Spielhallenklassiker neu aufgelegt: Taito präsentierte ein neues Breakout-Spiel namens Arkanoid. Verpackt wurde es mit einer neuen Spielstory: Irgendwo im Weltall suchen 4,5 Millionen Menschen — die Besatzung des Raumschiffes Arkanoid — einen neuen Planeten, ein neues Zuhause. Plötzlich wird das Schiff von außerirdischen Bösewichten schwer beschädigt. Ein Rettungsschiff — die Vaus — entflieht und wird mittels Mauern im Weltall gefangen. Der Spieler als Kommandant der Vaus, muß die Weltraummauern mit Energiebällen zerstören um das Schiff zu befreien. So sehr dies auch nach Weltraumschlacht klingen mag, die Breakout-Spielidee ist geblieben, auf dem Monitor heißt es Schläger und Ball gegen Mauer. Neu sind die Angreifer in Form von Würfeln und Kegeln, die auf dem Spielfeld herumfliegen und den Ball ablenken. Manche Mauersteine lassen beim Abräumen Kraftkapseln herunterfallen, die aufgefangen werden können. So bekommt man mehr Power: Einige Kapseln verlangsamen den Ball, vergrößern den Schläger, verleihen ein Extra-Leben, versehen den Schläger mit einem Laser etc.

Discovery Software aus Würzburg setzte den Automaten-Knüller als PAL-Version für den Amiga um. Während Computerumsetzungen von Automatenspielen oft enttäuschend sind, glänzt Amiga-Arkanoid durch Perfektion und Liebe zum Detail: Grafik, Sound, Spielablauf und die einzelnen Level sind identisch mit der Automaten-Version. Andere Breakout-Spiele für den Amiga können hier nicht mithalten. 32 knifflige Levels müssen überstanden werden, dann weicht Arakanoid vom Original ab: Discovery Software hat als Bonus noch zahlreiche weitere Level eingebaut — von einfach bis unmöglich reicht das Spektrum. Ist ein Spiel verloren, so kann man auf Wunsch weiterspielen, allerdings wird die erzielte Score auf Null gesetzt.

Arkanoid ist ein fesselndes Spiel, das Spielspaß auf lange Zeit garantiert und das man auch nach monatelanger Spielpause wieder aus der Schublade holt. Das Programm läuft auf allen Amiga mit 512 KB Speicher.

(mn)

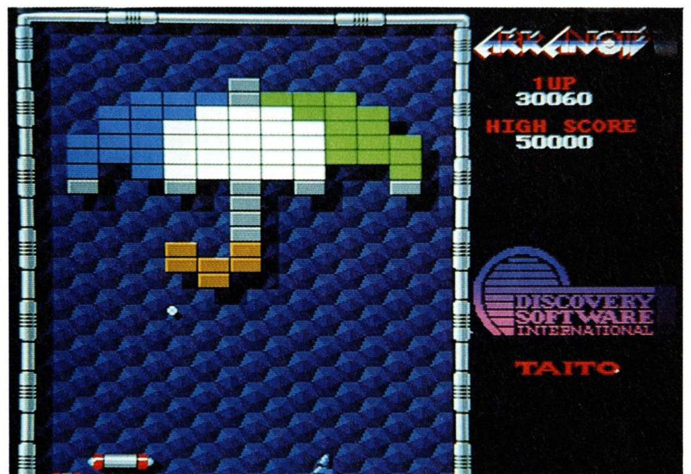
Name: Arkanoid
Preis: 79 Mark
Hersteller: Discovery Software International

Spaß	9
Grafik	10
Sound	8
Idee	7

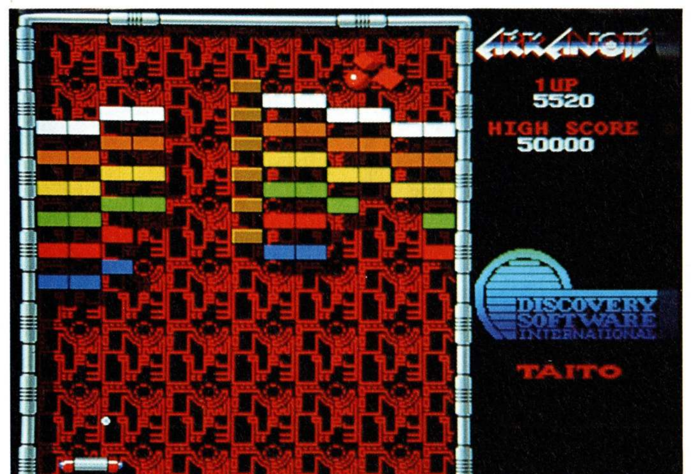
Schwierigkeit variabel



Kampfschiff „Vaus" verläßt die riesige „Arkanoid".



Die „Vaus" wirkt wie ein Schläger.



In manchen Mauersteinen sind Bonussymbole verborgen.

Kurz notiert:

Magic Bytes wechselte von MicroPartner zu Rainbow Arts. Die acht Programmierer und Designer sind für qualitativ hochwertige Spielsoftware bekannt. Von ihnen stammen feine Spiele wie „Western Games", „Pink Panther" und „Vampire's Empire". Welche der beiden Firmen in Zukunft das Label „Magic Bytes" führen darf, ist noch nicht entschieden.

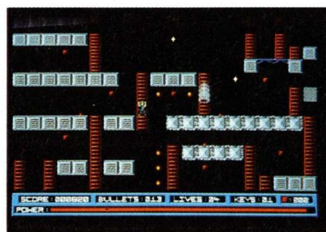
„Detector" heißt ein neues Ballerspiel von Timewarp Productions. Spieler werden zu Aliens und reisen von Planet zu Planet. Zwar ist die Grafik nicht gerade umwerfend, doch scrollt der Screen schnell und flüssig.

Tynesoft steckt Spieler in Boxershorts. Aber keine Angst ums Nasenbein, mit **„Seconds out"** wird nur am Bildschirm geboxt. Marco, das aufstrebende Boxtalent tritt gegen finster dreinblikkende Kontrahenten an. Es geht um die Weltmeisterschaft. Comicartige Atmosphäre holt den Spieler in den Kampf. Spaß ist garantiert.



Der gleiche Hersteller will Amokfahrer von der Straße an den Bildschirm locken. In **„Formula 1 Grand Prix"** können Hobbyrennfahrer ihre Fähigkeiten an fünf verschiedenen Edelflitzern testen. Sie wählen zwischen Ferrari F187, McLaren MP 4/3, Benetton B187, Lotus 997 und Williams FH 12 und suchen sich eine Rennstrekke aus. Rot — Grün — Gas! Wildes Schalten und riskante Überholmanöver sorgen für dramatische Augenblikke. Die Grafik könnte besser sein.

In **„Final Mission"** darf über Leitern und Fließbänder geturnt werden. Reline hat in der Vergangenheit Besseres geliefert! Im neuesten Produkt wird dem Käufer entsetzliche Grafik geboten.



„Space Ranger" von Mastertronic ist eine spaßige, grafisch hervorragende Rettungsaktion für bedrohte Planetenbewohner. Bei diesem Joystick-Abenteuer sieht man, was im Amiga steckt! Ein empfehlenswertes Vergnügen.

Auch in **„Galaxy Fight"** knallt es. Verschiedene galaktische Angreifer sind zu zerstören. Zu beziehen bei Müller.

Lieblinge der Götter verdienen ihr Geld nicht mit Handarbeit, nein, sie gewinnen es beim Lottospiel! Wer Fortuna nicht recht traut, läßt sich seine Gewinnchancen von **„Lotto Amiga"** berechnen. Auf Diskette sind alle Gewinnzahlen seit der ersten Ziehung 1955 gespeichert. Lottospieler können aus den Zahlen, die auf statistischen Werten basierend generiert werden, nach bestimmten Gesichtspunkten auswählen. Man setzt entweder auf Zahlen, die in den vergangenen Jahren besonders häufig oder nie gezogen wurden, oder auf Zahlen aus dem oberen, mittleren oder unteren Drittel der 49 Möglichkeiten. Nun kreuzt man seine auserwählte Zahlenkombination an, und das Programm klappert alle Gewinnzahlen ab. User bestimmen, in welcher Zeitspanne gesucht werden soll. Vielleicht hätten „ihre" Zahlen irgendwann einmal gewonnen. Wer seine Lottoliste auf dem neuesten Stand halten will, gibt jeden Mittwoch und Samstag die aktuellen Gewinnzahlen ein. Lotto-Fanatiker werden begeistert sein. Doch eins trübt die Freude: Die Druckerausgabe fehlt. Jeder Lottotip muß per Hand notiert werden. Für 34,50 Mark zu beziehen bei High Speed Software.

Diese Tippreihe wurde ermittelt! Testen Sie, was diese Reihe bisher erzielte durch Anklicken von Test!

Test	1	2	3	4	5	6	7
Ende	8	9	10	11	12	13	14
	15	16	17	18	19	20	21
	22	23	24	25	26	27	28
	29	30	31	32	33	34	35
	36	37	38	39	40	41	42
Tippreihen ermitteln	43	44	45	46	47	48	49

„Romantic Encounters at the Dome" ist eine Flirtschule für kontaktgeschädigte Computerfreaks. Flirtschüler müssen der englischen Sprache mächtig sein, um den rechten Charme zu erlernen. Die den Programmiererhirnen entsprungenen Partnerinnen werden sehr detailreich beschrieben; minderjährige Freaks könnten verwirrt werden. Micro-Illusions stellte das schmusige Programm her, vertrieben wird es von Activision/Ariolasoft.

„Rolling Thunder" spielt im New York der 60er Jahre. Spitzenagent Albatros soll eine Geisel aus der Gewalt des gegnerischen Geheimdienstes befreien. Laut Hersteller sind geboten „Intrigen und Heldentaten in einem intensiven Action-Spiel". Grafisch erhebt sich das Spiel nicht über C64-Niveau.



Digital Artwork, ein junges Programmiererteam aus Steinhagen, bringt frischen Wind in die Szene. Ihr Rollenspiel **„Dragonflight"** basiert auf einem guten Konzept und wurde perfekt illustriert. **„Master Soccer"** wird in einem halben Jahr erscheinen. Das Konzept erlaubt den Schluß, das es sich um eine Mischung aus „International Soccer" und dem legendären „Football Manager" handelt. **„Submarine Trouble"** heißt eine U-Boot-Simulation mit spektakulärer Handlung. Durch einen technischen Unfall gerät das U-Boot in ein Zeitloch. Nur ein einziger Wissenschaftler kennt den Weg in die Gegenwart. Doch der gute Mann scheint unauffindbar.

Die drei Spiele werden zuerst für Atari ST, C64 und PC erhältlich sein. Doch werden Amiga-User nicht vernachlässigt werden, verspricht Geschäftsführer Jens Müller.

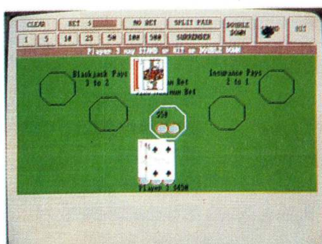
„Legend of the Sword“ heißt ein neues Grafik-Adventure von Silicon Software. Wird es ein Abenteuer im alten Britannien sein?

Activision wird **„Mindfighter“** für fast alle Systeme herausbringen.

PSS lädt zum Klassenkampf. In **„Power Struggle“** hetzen Spieler die beiden Großmächte aufeinander. Ein Staat soll dem anderen wirtschaftlich das Wasser abgraben und ihn mit militärischen Mitteln unter Druck setzen. Das Strategiespiel brilliert durch ethische Geschmacklosigkeit.



Wer gern ein Spielchen wagt, beweist sein Können in **„Black Jack Academy“**. Micro Illusions stellte das Kartenspiel sowohl für diejenigen her, die Black Jack bereits beherrschen, als auch für Lernwillige. Die Attraktivität liegt im Detail. Beispielsweise stehen Spielregeln nach Art verschiedener US-Städte zur Wahl. Ariolasoft vertreibt das Glücksspiel.



Actionware's Antwort auf „King of Chicago“ heißt **„Capone“**. Einer oder zwei Spieler räumen unter den Mafiosis auf. Hinter den Fenstern eines Chicagoer Viertels lauert die schießwütige Bande. Scharfschützen kommen von rechts und links in's Bild. Blitzschnell hat man die Feinde eliminiert; aus Versehen erwischt man auch mal einen Unbewaffneten. Doch scheinen die lebendigen Zielscheiben unter den Fensterbrettern nachzuwachsen. Wer vor nichts zurückschreckt, kann sich das Spiel bei Müller bestellen. 89 Mark.

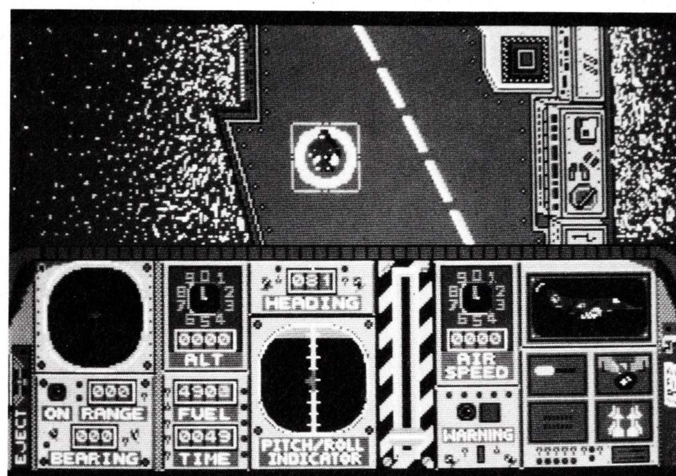


Bezugsquellen:
Amigaland, Hohenwaldstr. 26, 6374 Steinbach;
Atlantis Soft- und Hardware, Dunantstr. 53, 5030 Hürth 8;
CJS, An der Tiefenried 27, 3000 Hannover 1;
Elektronic von A—Z, Stresemannstr. 95, 1000 Berlin 61;
DTM, Bornhofweg 5, 6200 Wiesbaden;
Funtastic Computer Ware, Müllerstr. 44, 8000 München 2;
High Speed Software, W. Blanke, Postfach, 3362 Bad Grund/Harz;
Softwareversand Müller, Dorfstr. 1, 8852 Rain-Unterspeiching;
Pecksoft, Müllerstr. 44, 8000 München 2;
Profisoft, Sutthauser Str. 50/52, 4500 Osnabrück;
Seemüller, Schillerstr. 18, 8000 München 2;
Rushware, Bruchweg 128—132, 4044 Kaarst 2.

Für Testprogramme danken wir folgenden Firmen:
Activision
Ariolasoft
Leisuresoft
Profisoft
Rushware

Runter kommen sie immer

Mit „Jump Jet“ donnerten bereits C64-Besitzer über's nasse Element. Amiga-Fans werden ihre Flugkünste ebenfalls begeistert an dieser Mischung aus Jetsimulator und Arcadespiel erproben. Vier Spielstufen stehen zur Auswahl. In der ersten Stufe muß der Joystick-Pilot lediglich vom Flugzeugträger starten und ein paar Runden über dem Startareal drehen. Den Senkrechtstarter wieder sicher auf dem Flugzeugträger zu landen, ist dagegen ein kleines Kunststück. Und genau das wird in der zweiten Spielstufe erwartet. Endlich — in Stufe 3 — darf der Spieler mit seinem Düsenjäger in die Ferne rasen, wenden und den Anflug üben. Am spannendsten ist die rasante, letzte Spielstufe: Hier darf der Pilot auf die Tube drücken. Doch dieses Manöver ist nicht zum reinen Vergnügen des Fliegers gedacht. Er soll nämlich gegnerische Jets ausmachen und zerstören.



Nervenprobe: Landung auf dem Flugzeugträger

Dank einer vorbildlichen Anleitung gelingt es ihm bestimmt, wenn er sein Handwerkszeug — Joystick und Tastatur — beherrscht. Sämtliche Instrumente am Armaturenbrett werden erklärt, alle Tastaturkommandos aufgelistet. Wer sich ungeschickt anstellt, kann sich auf nasse Füße gefaßt machen. Versierte Flieger werden befördert. Als Flight Lieutenant beginnen sie ihre Laufbahn und arbeiten sich bis zum Group Captain empor. Mit diesen Möglichkeiten ist „Jump Jet“ realistisch. Auf allzu viele Details und Kinkerlitzchen verzichtete Anco jedoch, denn nicht nur Lufthansa-Kapitäne sollen hier Erfolgserlebnisse haben. Realistische Optik heißt nun aber auch — wie bei allen Flugsimulatoren — daß die Grafik recht triste ausfällt. Fürs Ohr gibt es eine digitalisierte Titelmusik, wirklichkeitsnahe Fluggeräusche und Funkdurchsagen.

(C. Borgmeier)

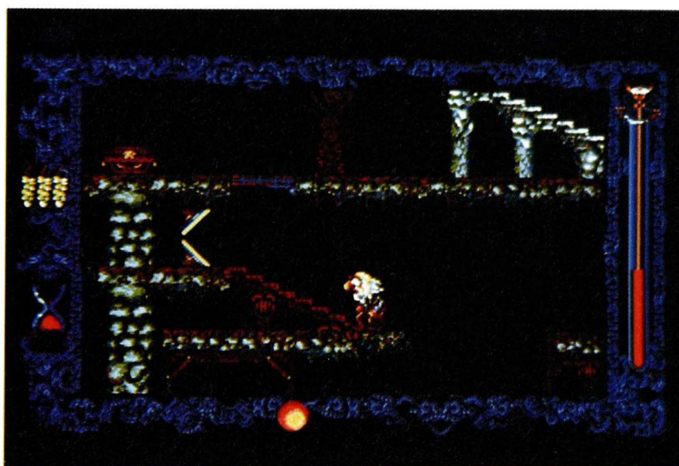
Name: Jump Jet
Preis: 49,95 Mark
Hersteller: Anco

Spaß	9
Grafik	6
Sound	8
Idee	7
Schwierigkeit	9

Knoblauch statt Kanone

Wer nicht aufpaßt, fühlt plötzlich einen schrecklichen Biß am Hals. Es ist der Fürst der Untoten, der seine langen Sauger in's Fleisch der Lebendigen schlägt. Dracula geistert wieder, und zwar im neuen, hervorragend animierten Renn-, Hüpf- und Taktikspiel „Vampire's Empire“. Doch dort, wo Bösewichte ihr Unwesen treiben, sind edelmütige Helden nicht fern. So auch in diesem Spiel. Der Held ist aus blutrünstigen Dracula-Filmen bestens bekannt, er heißt Dr. van Helsing, und seine Rolle ist dem Spieler zugedacht. Bewaffnet mit einer gehörigen Menge Knoblauch und Spiegel, bricht van Helsing zum Vampirreich auf. Mit den Spiegeln — so seine Strategie — wird er einen Sonnenstrahl durch die Grüfte leiten und Dracula töten. Über schier endlose Treppen geht es hinab. Durch mehr als 160 Bildschirme führt das Abenteuer. Ulkige, aber technisch verpatzte Sounds erklingen. Steinfratzen erschrecken van Helsing mit heftigen Niesern, bläuliche Schrumpfköpfe spucken von den Arkaden herunter. Doch auf der Suche nach dem blutsaugenden Adelssproß gibt es größere Probleme. Verfressene Ratten stürzen sich auf den Helden. Kräftige Tritte raffen die ekeligen Nager dahin. Aber vor allem Draculas geiferndes Vampirvolk interessiert sich für den appetitanregenden Eindringling. Der setzt jetzt das Allroundmittel Knoblauch ein. Die Knoblauchzöpfe müssen exakt an der Decke plaziert werden. Nur dann verbreiten die würzigen Knollen ihr intensives Aroma. Und die Plagegeister fallen reihenweise tot um. Trotzdem, man sollte sich nicht allzusehr auf die Wunderknolle verlassen. Jede Unaufmerksamkeit wird sofort bestraft, und der durstige Fürst labt sich am Heldenblut.

(Salus)



Opa van Helsing stolpert durch Draculas Fürstentum.

Name: Vampire's Empire
Preis: 60 Mark
Hersteller: Magic Bytes

Spaß	7
Grafik	8
Sound	5
Idee	7
Schwierigkeit	8

Drei Diamanten für Frankreich

Tips und Karte zu „The Three Musketeers“

Aufgabe: D'Artagnan muß einen Brief der Königin nach England bringen, dort drei Diamanten von Earl Bukkingham abholen und auf dem schnellsten Wege nach Paris zurückkehren.
Alle Aktionen werden mit Tastatur gesteuert. Antworten mit der Spacetaste auswählen und mit Return bestätigen. Himmelsrichtungen mit den Cursortasten eingeben. Werden schnelle Antworten gefordert (Sanduhr läuft), die Nummer der entsprechenden Antwort auf dem Zahlenblock eingeben.
Paris:

— D'Artagnan nimmt mit (3) den Brief aus Constances Händen entgegen;

— Geld, Brief und Papier nehmen und mit (3) vorsichtig aus dem Fenster klettern;

— den Passierschein gibt's bei De Treville, dem Chef der Musketiere; also nach (S) gehen;

— dem Sergeanten im Vorzimmer mit (4) Geld anbieten; mit (1) bestätigen;

— mit (1) die wichtige Mission wiederholen; mit (3) bestechen;

— bei De Treville: mit (3) vom Auftrag der Königin erzählen; Brief zeigen (3);

— Treue auf Ihre Majestät schwören (1); Treville ist überzeugt.
D'Artagnan muß nun seine Kameraden für den Auftrag gewinnen.

— (O) — (N) — (N) zum Hause des Porthos gehen; Porthos schläft fest;

— Wein trinken (3) und warten; warten (1);

— sobald er erwacht ist, von den bevorstehenden Abenteuern erzählen (1); Porthos ist überzeugt;

— nun zum Haus des Athos mit (O) — (S) — (S) — (W); der sitzt benommen dort;

— versuchen, ihn aufzurütteln (1);

— ein Glas Wasser in's Gesicht schütten (2); mit (1) überreden; Athos macht mit.
Nun fehlt nur noch Aramis;

— sein Quartier liegt (N) — (W) — (W) — (S) — (S);

— Aramis schreibt gerade Liebesbriefe; (1) überzeugt ihn davon, daß es Wichtigeres gibt;

— die Frage nach den Papieren mit (2) beantworten; mit (1) bestätigen; Aramis macht sich reisefertig.

— (N) — (N) führt zum Stadttor von Paris;

— dank der Papiere dürfen die vier passieren (W).

— vor der Stadt eine Taverne betreten (N);

— mit (2) einen Betrunkenen abwehren;

— einem aufdringlichen Bettler mit (3) Kupfermünzen zuwerfen;

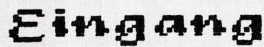
— mit (2) gegen den Hinterhalt der Richelieu-Anhänger verteidigen;

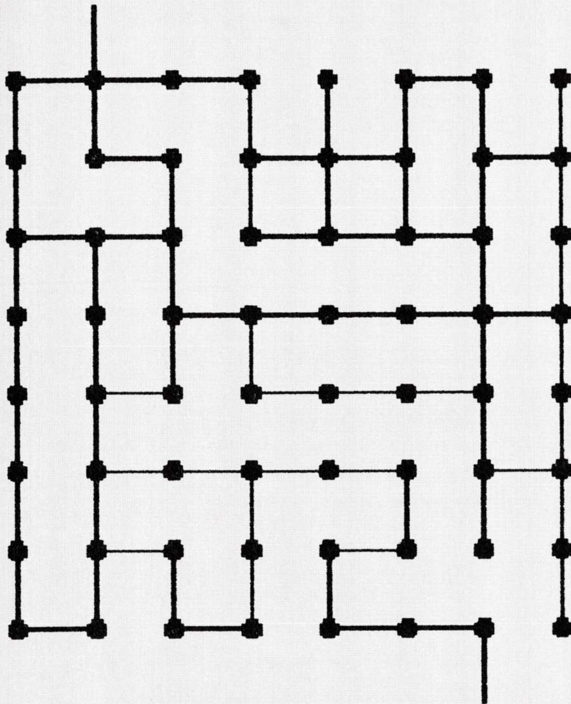
— weitergehen nach (N) — (O) — (S); der Eingang des Waldes = Labyrinth ist erreicht.

Der schnellste Weg durch den Wald führt nach (S) — (O) — (S) — (S) — (W) — (S) — (O) — (O) — (O) — (O) — (S) — (W) — (S) — (O) — (O) — (S).

Ab hier sollen Adventure-Fans selber weiterknobeln.

(Salus)





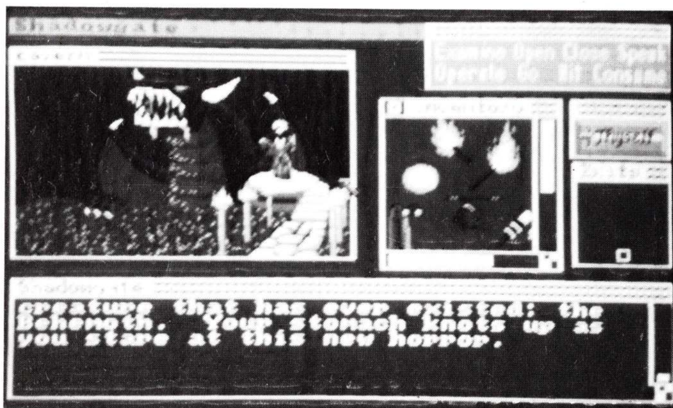
Karte des labyrinthischen Waldes

Schatzsuche mit Giana

„The Great Giana Sisters“ birgt eine Menge Überraschungen, die man beim flüchtigen Durchspielen nicht zu sehen bekommt. Giana-Programmierer Armin Gessert kramt in seiner Trickkiste: In einigen Schluchten ist ein blauer Fels zu sehen. Springt Giana dort hinein, landet man in einer Schatzkammer, wo man ohne Gefahr Diamanten einsammeln kann. Die erste Schatzkammer befindet sich in Level 5.
Ein Feature, das ebenfalls nicht in der Anleitung erwähnt wird, ist die „Warp Zone“. Wenn man unter bestimmte Steine springt, kommt Giana in einen wesentlich höheren Level. Die erste Warp Zone befindet sich in Level 3 unter einem Stein. Springt Giana darunter, wird sie in Level 6 teleportiert. In Level 6 befindet sich über der Brücke ebenfalls eine Warp Zone, mit der man in Level 12 gelangt. In Level 12 ist eine weitere Warp Zone. Sie ist unter einem Stein in einer Kammer unter der Decke zu finden.
Drachen und Spinnen darf man nicht abschießen. Bei jedem Treffer stürzen sich die bösen Tiere auf Giana, so daß man ein Bildschirmleben verliert. Um den Drachen zu passieren, braucht man nur unter ihm durchzulaufen. Über die Spinne springt man hinweg.
(cb)

Tips und Tricks zu „Shadowgate“

- Öffne den Globus mit Hilfe der Papierrolle „Land under the heavens; the key to the world — Terra Tarrak“. So erhältst Du einen weiteren Schlüssel und einen Zaubertrank, der neue Kraft verleiht;
- öffne den Stein im Laboratorium mit der Öse; es erscheint eine Flasche mit der Aufschrift „Heiliges Wasser“;
- die Gargoyles können nur ein einziges Mal passiert werden, deshalb untersuche zuerst den Cavern-Room. Dort gibst Du der riesigen Statue das schwarze Buch (operate — book — statue); eine Brücke erscheint;



Von wegen Fabelwesen! In „Shadowgate“ werden Legenden lebendig und bedrohen Eindringlinge.

- der Eisenzylinder kann nur mit richtiger Hebel-Kombination geöffnet werden; die Kombination lautet: 3 — 2 — 3; der Zylinder öffnet sich und „Silver Orb“ erscheint;
- beim „Platinum Horn“ öffnest Du zuerst die Flasche mit heiligem Wasser und nimmst das „Platinum Horn“ mit;
- im Observatorium nimmst Du den Stern von der Sternenkarte mit, zündest die Karte an, und es erscheint ein Stab; damit gelangst Du zum Zauberstab auf dem Balkon;
- das „Golder Thorn“ erlangst Du, wenn Du die Frau (die in Wirklichkeit ein Werwolf ist) im Turmschlößchen mit dem kleinen Pfeil erledigst;
- töte den Drachen mit dem Stern, und Du bekommst die „Blended Sun“;
- blende die Gargoyles kurz mit der Papierrolle „To move the sun from far to near: Instantum Illuminaris“ und geh vorbei;
- öffne den Brunnen mit der Hand; wirf die große Goldmünze hinab, die Du in einem Beutel im Aussichtsturm findest; steig in den Brunnen;
- Du stehst am Fluß. Schlag auf den Gong; der Fährmann mit seinem Floß rudert heran. Gib ihm eine Münze und steig ein. Plaziere die „Blended Sun“ ganz links und blase anschließend kräftig in's „Platinum Horn“.

Nun mußt Du nur noch herausfinden, wozu Du die „Silver Orb“ brauchst, und wie Du das große Ungeheuer beim Zauberer erledigst. Viel Glück!

(Salus)

Zombies im Irrgarten

Lösungstip zu „Uninvited“

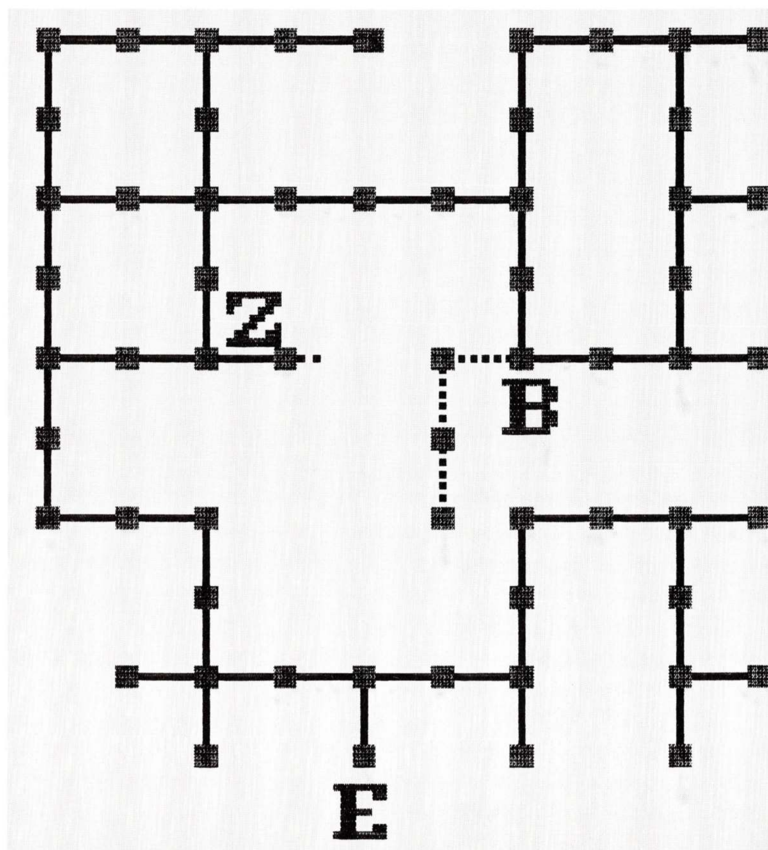
E .... Eingang

Z .... Zombies

B .... dort den Blumenstrauß am Grabstein niederlegen, und der Geheimgang öffnet sich.

Dann die drei Käfige öffnen und den Vogel in den eigenen Käfig sperren. Weiter nach Süden gehen bis zum springenden Monster. Hier den Vogel wieder freilassen, das Monster folgt ihm. So kann man nun problemlos den Edelstein aufnehmen. Die Gruppe Zombies kann man nicht vernichten. Einzelne Zombies kann man mit dem Amulett verjagen.

F. Wallner

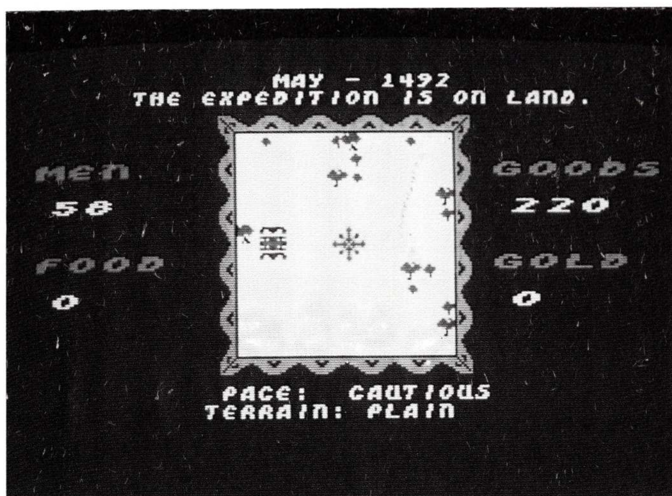


Endlich eine Wanderkarte für's Labyrinth!

Suche nach El Dorado

„Mein König! Ich werde aufbrechen und für Euch eine neue Welt entdecken. Es soll eine Welt voller Geheimnisse sein, die unermeßliche Reichtümer birgt.“ So mag im 15. Jahrhundert ein tollkühner Kapitän zu seinem Herrscher gesprochen haben. Welche Majestät würde einem so vielversprechenden Untertan ihren Segen verweigern? Doch darf der Kapitän erst zurückkehren, wenn er der Krone Gold und Ruhm bringen kann. Kapitän K. verläßt den Palast mit einem mulmigen Gefühl. Wird er sein voreiliges Versprechen halten können? Solche Bedenken hat der Computer-Conquistador nicht. Joystick in Port 0, und los! Man steuert nun seine Spielfigur — diese grafische Verunglimpfung eines menschlichen Wesens — durch's Hafengelände. Im Haus des „Outfitters“ werden Proviant und Handelsgut abgecheckt. Alle Vorbereitungen sind getroffen. Und Kapitän K. geht an Bord. Szenenwechsel. Der Spieler navigiert jetzt einen kleinen Schoner durch den Ozean und beobachtet seine Manöver in einem zu klein geratenen Bildschirmfenster. Darin ist Scrolling in acht Himmelsrichtungen möglich. Steht der Wind günstig? Konstanter Ostwind bläst in die Segel. Endlich erspäht Käpt'n K. Land. Wer fix im Geschichtsbuch nachgeblättert hat, schlußfolgert, daß es sich um die Bermuda-Inseln handeln müßte, die Kolumbus 1492 entdeckte und fälschlicherweise für Westindien hielt. Am Bildschirm entdeckt der Spieler lediglich gelbe Flecken im Blau des Ozeans. Neugierig, wie Entdecker sind, geht er von Bord. Zu Fuß werden ausgewählte Inseln erforscht. Rothäutige Eingeborene müssen abgewehrt werden. Bäche und Flüsse, die kein Weißer vorher zu Gesicht bekommen hat, werden überquert. Und endlich kann sich Kapitän K. seinem Vaterland nützlich erweisen, indem er indianische Goldschätze plündert. Mit seinem Schiff voll edlem Metall kehrt er auf schnellstem Wege in die Alte Welt zurück, um seinem König Goldklumpen zu Füßen zu legen. In majestätischer Großmut bedeckt ihn der Gekrönte mit Ruhm und Titeln. Weniger mit Ruhm bekleckert haben sich dagegen die EA-Grafiker und Soundprogrammierer. Mag die Aufmachung einen C64-Besitzer beglückt haben, so erwarten Spieler von der Amigaversion mehr. Ausgleich schafft ein abenteuerliches Ambiente, das den Spieler aus dem Atomzeitalter direkt in's späte Mittelalter entführt.

(cb/ub/mss)



Nirgends entdecken die goldgierigen Eroberer gekreuzte Palmen. Wo liegt denn hier der Schatz?

Name: The Seven Cities of Gold
Preis: 60 Mark
Hersteller: Electronic Arts

Spaß	7
Grafik	1
Sound	1
Idee	9
Schwierigkeit	8

Londons feinste Nase

London, Ende des 19. Jahrhunderts. Krumme Gassen, klappernde Hufe. Dichter Nebel dringt durch den Mantel. Das ist das Ambiente eines neuen Detektiv-Adventures von Infocom.

Sherlock Holmes' jüngster Fall ist die Suche nach den unbezahlbaren Kronjuwelen. Anläßlich eines Jubiläums will Königin Viktoria der Öffentlichkeit ihre berühmten Kleinodien zeigen. Doch wird das Volk auf den glanzvollen Anblick verzichten müssen, da ein ausgekochter Dieb die Juwelen gestohlen hat. Trotzdem soll die Präsentation um 9 Uhr stattfinden. Es ist 5 Uhr morgens, als Dr. Watson, Freund und Arzt der berühmten Spürnase, in die verträumte Bakerstreet einbiegt. Da schlüpft der Spieler in die Rolle des Arztes. Die Zeit drängt. Holmes und Watson begeben sich sofort auf Indiziensuche und werden fündig. Aha, der Dieb ist raffiniert und schalkhaft! Er hat doch tatsächlich einen Zettel mit Versen am Tatort hinterlassen! Holmes und Watson können sich noch keinen Reim darauf machen. Aber den beiden Detektiven fällt bestimmt etwas ein. Sie haben schließlich einen guten Ruf zu verlieren. Infocom schraubte die Anforderungen allerdings nicht zu hoch. Niemand möge sich abschrecken lassen. Der Täter selbst gibt versteckte Hinweise, und das Programm verrät einige Tricks. Daß der Fall nichts für Schlafmützen ist, vermittelt die eingeblendete Uhrzeit. Bis 9 Uhr müssen die Glitzersteine aufgetaucht sein. Das Programm arbeitet glücklicherweise nicht in Echtzeit. „Sherlock Holmes“ ist neben „Hitchhiker“ das einzige Programm, zu dem Infocom keine eigene Geschichte lieferte. Nachdem das Roman-Material bearbeitet ist, bietet der Text immer noch reichhaltig Londoner Impressionen verbunden mit einem Einblick in's Metier der Kombinierens und Spurenlesens.

(ub/R. Goldmann)

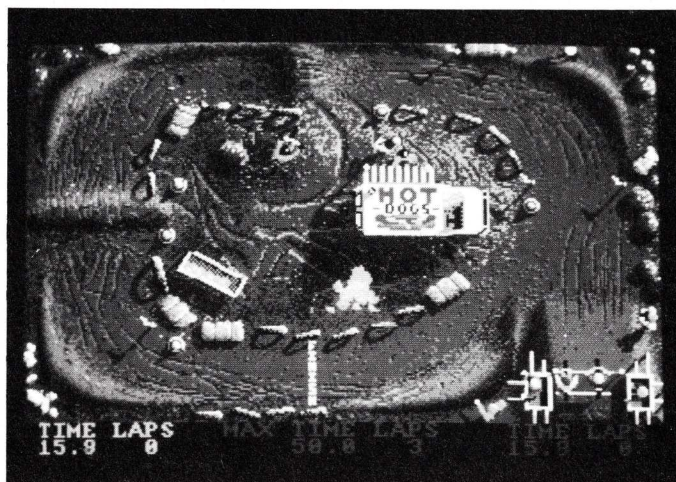
Name: Sherlock Holmes and the Riddle of the Crown Jewels
Preis: 90 Mark
Hersteller: Infocom

Spaß	7
Parser	10
Sound	keiner
Idee	6
Schwierigkeit	8

Starkes Profil

„BMX Simulator" begeistert durch interessante Spielfeatures. Fahrradfreunde betrachten aus der Vogelperspektive bunte, detailliert dargestellte Bahnen voller Hindernisse. Sieben warten darauf, überwunden zu werden. Der erste Parcours — eine kreisrunde Rennbahn — macht wenig Probleme. Nach jedem Unfall kann man wieder aufstehen und weiterstrampeln. Erst dann wird's schwierig. Scharfe Kurven erhöhen die Unfallgefahr, Engpässe fordern Höchstleistungen. Hier und da sind Sprungschanzen sichtbar. Sie bleiben jedoch ohne Wirkung, so als sei nicht die geringste Erhebung vorhanden. Ein oder zwei BMX-Artisten können sich auf die Piste begeben. Während der Fahrt finden sie heraus, wo sich technische Macken verbergen. Bereits im zweiten Parcours werden sie fündig. Berührt ein Fahrer die Strohballen am Pistenrand, ist das Rennen für ihn gelaufen. Er stürzt fast über den Bildschirmrand hinaus, steht auf und stürzt wieder. Aus diesem Teufelskreis kommt er nicht mehr heraus. Der Konkurrent darf sich seines Sieges sicher sein. Allerdings darf er die vorgeschriebene Qualifikationszeit nicht überschreiten. Wer allein spielt, ärgert sich über die vielen Kurven. Vor jeder Kurve muß er viel zu stark abbremsen und sich auf gerader Strecke ordentlich in's Zeug legen. Nur so kann er seinen computergesteuerten Gegner, der mit fast unverminderter Geschwindigkeit vorbeiradelt, überholen. Nach der Auswertung schaut man den mehr oder weniger erfreulichen Rennverlauf noch einmal per Action-Replay-Option an. Dabei fällt auf, daß „BMX-Simulator" viele Gemeinsamkeiten mit Activisions „Super Sprint" aufweist.

(cb)



Off-Road-Gestrampel aus der Vogelperspektive.

Name: BMX Simulator

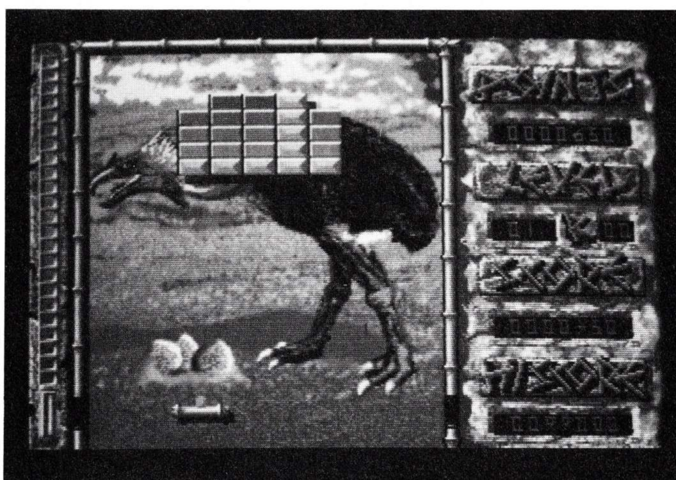
Preis: 35 Mark
Hersteller: Codemasters

Spaß	4
Grafik	8
Sound	7
Idee	3
Schwierigkeit	8

Zurück in die Steinzeit

Breakouts sind nicht auszurotten. Linel präsentiert nun eine steinzeitliche Version der Mauerknallerei: „Crack". Wie in jedem Arkanoid-Verschnitt steuert der Spieler mit der Maus einen Schläger am unteren Bildschirmrand und bricht mit einer Kugel Steine aus unterschiedlichen Mauerformationen. Einige Steine enthalten Bonussymbole, die Kugel oder Schläger vorübergehend mit neuen Eigenschaften versehen. Klebeeffekt, eingebaute Kanonen, schmale und breite Schläger machen die Steinbrucharbeit abwechslungsreich. Darin gleicht ein Breakout dem anderen. In der Variante „Crack" sollen unter dem Thema Steinzeit frische Zutaten in den Breakout-Einheitsbrei gestreut werden. In der Titelsequenz schwingt ein Steinzeitmensch an einer Liane ins Bild und stößt markerschütterndes, digitalisiertes Tarzangeschrei aus. Kurz darauf rennen drei seiner Artgenossen herbei und meißeln den Spieltitel aus dem Fels. Gedämpfte Bongoklänge und digitalisierte Urlaute verschmelzen zu einem fantastischen Sound. Abrupt beginnt der Breakout-Alltag. Im Gegensatz zu anderen Versionen aus dieser Spielefamilie hat Linel an eine Zwei-Spieler-Option gedacht und für vorgeschichtliche Atmosphäre gesorgt. Hinter langweiligen Backsteinwänden verbergen sich schuppige Saurier, langzahnige Mammuts, häutige Flugdrachen und andere Kreaturen, deren Vorbilder wir lediglich aus Knochenfunden und Versteinerungen kennen. Das motiviert nicht nur Geologen und Evolutionsforscher! Linel entzückt destruktive Mauerbrecher aber nicht nur mit grafischen Kostbarkeiten, sondern läßt sie am Screeneditor kreativ werden. Mit dem Editor können sie Wände nach eigenen Vorstellungen mauern und hinterher zerknallen.

(cb)



Urzeitliches Getier wird langsam freigelegt.

Name: Crack

Preis: 65 Mark
Hersteller: Linel

Spaß	8
Grafik	9
Sound	9
Idee	1
Schwierigkeit	6

Freud und Leid in der Steinzeit

„Japadapaduu“ schreit Vater Feuerstein aus voller Kehle. Feierabend! Fred freut sich auf's abendliche Bowling mit Barney. Doch er hat sich zu früh gefreut, denn Wilma hat Arbeit für ihn. Fred soll die Wände des Felsenheims streichen und nebenbei auf Töchterchen Pebbles aufpassen. Wie nervenaufreibend dieser Doppeljob ist, stellt sich nach kurzem Spiel heraus. Fred muß nicht nur sauber pinseln, sondern auch verhindern, daß Pebbles die frisch gestrichenen Wände bekritzelt. Ist diese Arbeit zu Wilmas Zufriedenheit erledigt, rasen die beiden Freunde im Steinzeitflitzer zum Bowling-Center. Da wetteifert Fred mit Kumpel Barney. Kein Problem für Fred. Barney wird zwar vom Computer gesteuert, aber Feuerstein hat ein gutes Augenmaß, landet einen wohlgezielten Schub nach dem anderen . . . und siegt. Doch es bleibt kaum Zeit zur Freude, denn Pebbles schwebt inzwischen in höchster Gefahr. Ob Fred die Kleine retten und irgendwann einmal verschnaufen kann, entscheidet sich am Joystick. Das hört sich aufregend an, ist es aber nicht. „Fred Feuerstein“ ist ein mittelmäßiges Spiel, denn es fehlen spielerische Herausforderungen. Vielmehr freuen sich Flintstone-Freunde über das Wiedersehen mit den vertrauten Trickfilmfiguren aus dem ulkigen Bedrock County. Da gibt es einiges zu schmunzeln: Neues vom Tage erfährt die steinzeitliche Familie aus steinernen Fernsehern mit Knochenantennen, prähistorische Gespräche werden in Sprechblasen wiedergegeben. Insoweit erkannten die Grandslam-Programmierer den einzigartigen Witz der Kultserie und griffen einige Features auf. Ariolasoft legt dem Programmpaket ein Feuerstein-Poster und eine Anstecknadel bei.

(Matthias M. Schmidt)



Von wegen Feierabend: Babysitting und Malerarbeiten strapazieren Fred's Nervenkostüm.

Name: Fred Feuerstein
Preis: 60 Mark
Hersteller: Grandslam

Spaß	4
Grafik	6
Sound	7
Idee	9
Schwierigkeit	3

Kürbis im Farbenrausch

„Wizball“ ist ein C64-Bestseller. Die Amiga-Version hat gute Chancen, ebenfalls ein Treffer zu werden. Aufbau und Story sind unverändert. Wie gehabt, gleiten faszinierte Spieler hinüber in die fremdartige Wizwelt am anderen Ende der Milchstraße. Zauberer Zark verwandelte den ehemals kunterbunten Planeten Wiz in eine triste, graue Himmelskugel. Irdische Joystickakrobaten sollen die öden Landschaften wieder in alter Farbenpracht erstrahlen lassen. Mit dem Steuerknüppel wird ein flummigleich springender Kürbis über's trostlose Terrain manövriert. Hindernisse schränken die Bewegungsfreiheit ein. Zu allem Überfluß greifen Scharen außerirdischer Feinde an.

Da kommt der Feuerknopf nicht zur Ruhe. Extrawaffen wie Laserstrahler oder Antigravitationsspray helfen bei der Verteidigung. Trotz ständiger Sorge um das Kürbisleben muß man nebenbei rote, grüne und blaue Farbballons abschießen und herausperlende Tropfen auffangen. Die Farben werden in Kesseln am unteren Bildschirmrand gesammelt. Und sobald alle Farbtöpfe eines Levels gefüllt sind, ist ein Achtel des Planeten wieder bunt und lustig anzusehen.

Durch Vulkankrater springt der Kürbisball in die nächste Spielebene und bemüht sich wiederum um farbige Landschaften. Sobald die Joysticksteuerung perfekt beherrscht wird, kommt niemand mehr vom „Wizball“ los, ehe die triste Wizwelt nicht wieder in freundliches Licht getaucht ist.

An Motivation fehlt es bestimmt nicht; weder an der Grafik noch an der Spielidee gibt es etwas auszusetzen. Nur mit Sounds wurde geknausert. Nach einer erfreulichen Titelmelodie vernimmt der Spieler lediglich primitive Spielgeräusche. Doch das läßt sich verschmerzen. Lediglich der hohe Preis stört. *(cb)*



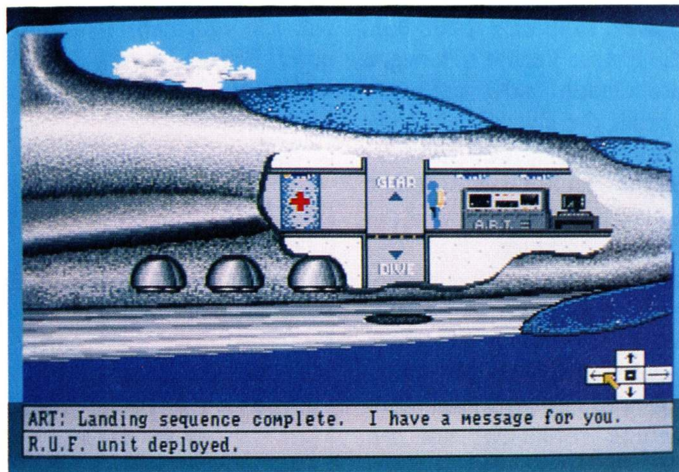
Wizball soll seine Welt in neue Farben kleiden.

Name: Wizball
Preis: 80 Mark
Hersteller: Ocean

Spaß	9
Grafik	8
Sound	6
Idee	10
Schwierigkeit	9

Atlantis: versunkenes Geheimnis

Seit Menschengedenken fasziniert der Mythos Atlantis. Im 5. Jahrhundert vor Christus wird der sagenhafte Inselstaat erstmals von Plato erwähnt. Charles Berlitz, heute erfolglos. Oder besser: bis gestern, denn heute bietet Electronic Arts zumindest die Illusion eines siebenten Kontinents am Bildschirm. Bevor nach dem versunkenen Landmassiv gesucht werden kann, sind 13 Missionen in festgelegter Reihenfolge zu erfüllen. Da

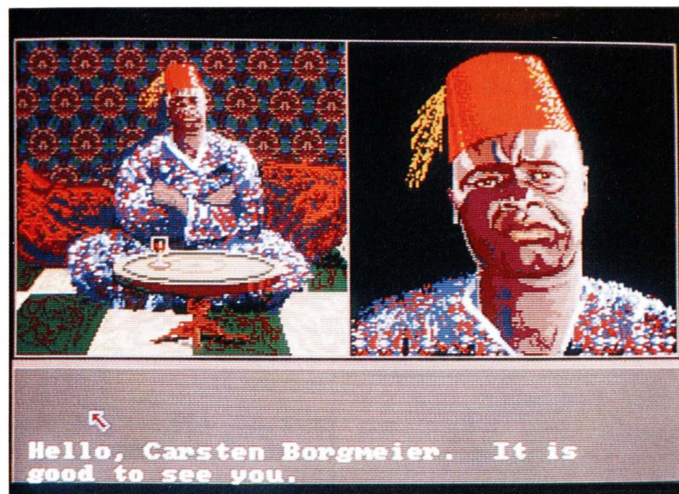


Hightech-Schiff „Viceroy" ist für alles gerüstet.

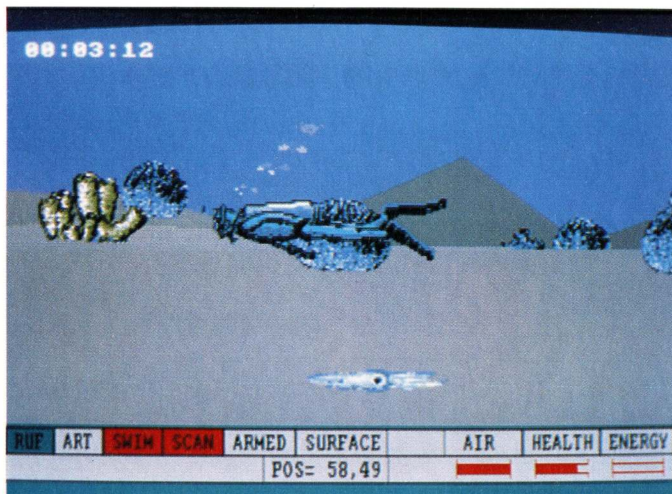
Geheimnisforscher unserer Tage, vermutet, daß vor 2500 Jahren nicht nur Elektrizität, sondern sogar die Kunst des Fliegens bekannt gewesen sei. Eine Katastrophe setzte dem ganzen ein Ende; der Kontinent versank im Ozean. Seitdem forschen Abenteurer jeglicher Couleur nach Resten der einstigen Hochkultur. Bis mag sich zwar mancher Atlantis-Fan geprellt fühlen, doch den Spielspaß beeinträchtigt dies nicht. Wirklich frustrierend ist dagegen der ständige Diskettenwechsel bei einem einzigen Laufwerk. Von wegen Spielaction! Zwei Laufwerke machen mobiler. Eine animierte Geistermaske erklärt kurz den neuen Agenten-Job. Sie verrät, daß von den Perilia-Insulanern Hinweise auf Schätze und Ortsangaben zu bekommen sind. Unser Agent muß raffinierte Fragen stellen, um die Eingeborenen zum Reden zu bringen. Alles, was er herausbekommt, sind verschlüsselte Informationen. Nun liegt es bei ihm, Aussage für Aussage so zu interpretieren, daß sie bei der Erfüllung der Aufträge weiterhelfen. Mit der futuristischen „Viceroy" geht Agent X in See. Neben einer standardisierten Ausstattung mit Bordcomputer, Lift und Kommandobrücke verfügt das Expeditionsschiff über die wahrhaft utopische Einrichtung einer Heilungszelle; Verletzungen verschwinden darin binnen kürzester Zeit. So ist selbst für den schlimmsten Unfall vorgesorgt. Damit es erst gar nicht dazu kommt, setzt man sich auf Tauchgängen mit dem Bordcomputer in Verbindung. Der elektronische Helfer gibt über Funk Neuigkeiten und gute Ratschläge, dazu Daten über die Beschaffenheit des Meeresbodens. Speziellen Unterwasser-Jobs dient ein programmierbarer Roboter. Er ist in der Lage, gewünschte Stoffe aufzuspüren und allerhand Nützliches zu erledigen. Unter diesen Voraussetzungen sollte es ein Leichtes sein, alle Missionen zu erfüllen und allen Widrigkeiten zu begegnen. Und Gefahren gibt es wirklich genug. Monströse Tiefseekraken und menschenfressende Raubfische bedrohen das Leben. Kerle, die in ihrer Gier nach Reichtümern vor nichts zurückschrecken, stellen sich gegen den Agenten. Er löst all diese Probleme mit seiner Intelligenz und, falls sie nicht ausreicht, mit Waffengewalt. Diese Features sind es, die „Return to Atlantis" zu einem sehr komplexen und sehr interessanten Spiel machen. Grafik und Sound erfüllen allerdings nicht die Anforderungen, die man inzwischen an EA-Produkte zu stellen gewohnt ist.
(cb)

Name: Return to Atlantis
Preis: 130 Mark
Hersteller: Electronic Arts

Spaß	8
Grafik	5
Sound	5
Idee	6
Schwierigkeit	6



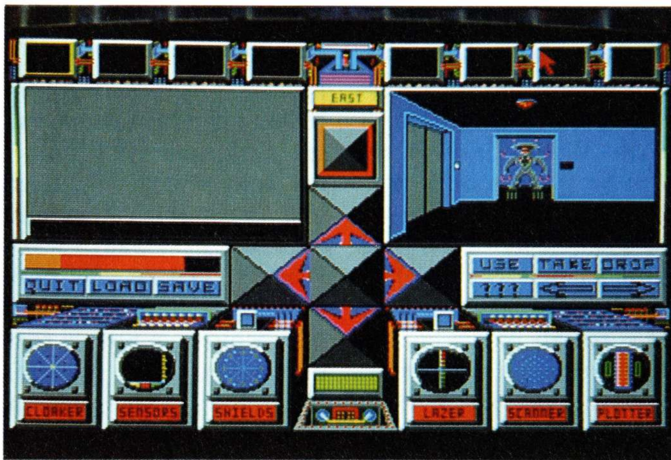
Agenten brauchen gute Freunde in aller Welt.



Die stille Unterwasserwelt birgt zahllose Geheimnisse und Gefahren.

Öder Irrgarten

In geheimer Mission schleicht Slaygon, ein hochentwickelter Kampfroboter, durch die Gänge von Cybordynamics Industries. Der Konzern arbeitet an einem giftigen Virus, der die gesamte Erdbevölkerung dahinraffen soll. Slaygons Job ist es, das Desaster zu verhindern. Im weitverzweigten Netz von Gängen, Türen und Laborräumen sucht er nach fünf Codes. Sobald man diese Kombinationen eingibt, schaltet der Hauptcomputer das Kühlsystem des Reaktors aus. Bevor nun der überhitzte Reaktor explodiert, muß Slaygon sich schnellstens aus dem Staub machen.



Glutäugige Roboter fordern zum Zweikampf.

Die Geschichte liest sich wie das Drehbuch zu einem neuen Actionfilm; deren Umsetzung ist jedoch weniger glanzvoll. In einem Labyrinth, in dem alles gleich aussieht, fast alle Türen verschlossen sind und wenige gegnerische Roboter genügen, einem das Licht auszublasen, stellt sich schon bald die Frage: Verlier' ich zuerst die Geduld oder geht mir vorher die Energie aus? Die wenig abwechslungsreiche Suche nach Gegenständen, Energie und Codes wird dadurch erschwert, daß Cybordynamics aus vielen kleinen Labyrinthen besteht. Wozu die darin herumliegenden Gegenstände benötigt werden, wird in der Anleitung (Englisch) nur andeutungsweise — wenn überhaupt — erklärt. Letztes Manko ist die zu klein geratene Labyrinthkarte.

Vorteilhaft: Ein Spiel kann unterbrochen und auf Diskette abgespeichert werden, um so immer tiefer im gleichen Labyrinth vorzudringen. Ansonsten errechnet der Computer für jedes Spiel eine neue Labyrinthkonstruktion.

Slaygon ist wohl eher etwas für Geduldsmenschen mit einer Schwäche für Labyrinthe als für Actionfreunde.

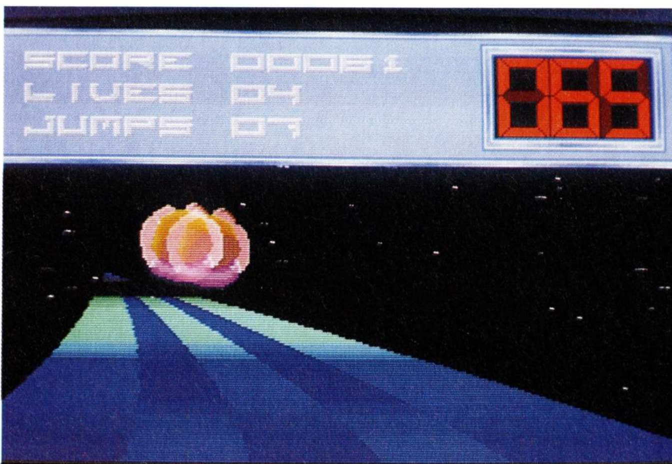
(Matthias M. Schmidt)

Name: Slaygon
Preis: 65 Mark
Hersteller: Microdeal

Spaß	3
Grafik	5
Sound	2
Idee	8
Schwierigkeit	10

Teure Hopserei

Robtek, bekannt durch schwache Amigaspiele wie „Rocky", „Kwasimodo" und „Ball Raider" wirft einen neuen Flop auf den Softwaremarkt: „Starways". Wie enttäuscht muß der Käufer sein, wenn er feststellt, daß es sich bei „Starways" nur um einen schwachen Aufguß vom C64-Spiel „Trailblazer" handelt. Man hoppelt mit einem Männchen über eine Rennstrecke, die aus farbigen Rechtecken zusammengesetzt ist. Jedes Feld birgt andere Schikanen, mit denen Spieler zurechtkommen müssen. Da gibt es braune Felder, die das rasante Tempo der Witzfigur reduzieren, grüne und rote, die es Saltos und Luftsprünge vorführen lassen. Am gefährlichsten sind schwarze Felder. Sobald der Läufer ein solches Feld überrennt, stürzt er in die Tiefe des Weltraums. Also aufpassen!



Gefährliche Tour über löcherige Weltraumpisten

Mit einem Joystick in der Hand versucht der Spieler, das Männchen über die Piste zu jagen, ohne in schwarze Löcher zu fallen. Das ist leichter gesagt als getan, da man nur siebenmal pro Streckenabschnitt hopsen darf. Gerät man an ein schwarzes Feld und hat seine sieben Sprünge im Spielabschnitt bereits vertan, ist eins der raren Bildschirmleben futsch. Auf die Dauer ist das ganz schön frustrierend. Erst wenn man die Strecke im Schlaf beherrscht, steigen die Erfolgsaussichten. Grafisch läßt sich „Starways" als erträglich bezeichnen. Die 3D Grafik ist schnell; über die Animation des Männleins kann man ebenfalls nicht klagen. Doch von Läufer und Piste einmal abgesehen, sieht es sehr trostlos auf dem Bildschirm aus. Apropos trostlos: Eine kurze, sich ständig wiederholende Titelmelodie überreizt die Gehörnerven. Digitalisieren kann inzwischen jeder Seppel. In einem Spiel für 65 Mark darf man programmierte Musik erwarten.

(cb)

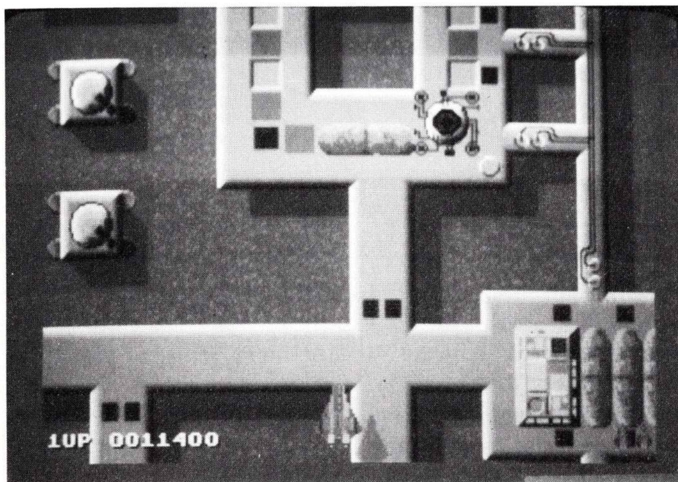
Name: Starways
Preis: 65 Mark
Hersteller: Robtek (Diamond Software)

Spaß	2
Grafik	6
Sound	2
Idee	1
Schwierigkeit	9

Brillante Ballerei

Ein gigantischer Sternenkreuzer bedroht das Universum. Die Besatzung wäre in der Lage alles Leben zu vernichten. Nur ein todesverachtender Jagdflieger kann die außerirdische Gefahr besiegen. Er wagt sich tollkühn in das Innere des Kolosses und will alle Verteidigungsanlagen zerstören. Für die Hintergrundstory von „Sidewinder" hätte sich Mastertronic Originelleres einfallen lassen können! Böse Aliens treiben in nahezu allen Actionspielen ihr Unwesen. Doch ausschlaggebend sind spielerische Qualitäten. Und davon hat das Weltraumgeballer einiges zu bieten. Vier Sektoren des feindlichen Raumschiffes müssen überwunden werden. Jeder Abschnitt ist mit unterschiedlichsten Geschützanlagen bestückt. Aber Vorsicht! Scharen gegnerischer Kampfschiffe verteidigen das martialische Mutterschiff. Trotzdem will unser Pilot seine höllische Mission erfüllen, nämlich die Zentraleinheit des Schiffes zu zerstören. Computerpiloten schießen, die Finger um den Joystick gekrampft, auf gegnerische Geschütze und simple Raumgleiter. Pchhhhh! Bei jedem Treffer erhellen entsetzliche Explosionen den Bildschirm. Perfekt animierte Kampfroboter mischen sich unter das Heer der Feinde. All diese gut durchdachten und ebensogut realisierten Accessoires bringen Bewegung in's Spiel! Hier entstand aus einer primitiven Spielidee kein ödes Programm. Feiner Titelsound und abwechslungsreiche Grafik verbunden mit hervorragender Spielbarkeit motivieren. Fünf verschiedene Schwierigkeitsstufen fordern heraus. Von Anfang an müssen sich die Spieler ordentlich anstrengen.

(cb)



Sidewinder auf dem Weg zur Zentraleinheit.

Name: Sidewinder	
Preis: 35 Mark	
Hersteller: Mastertronic	
Spaß	10
Grafik	9
Sound	8
Idee	3
Schwierigkeit	10

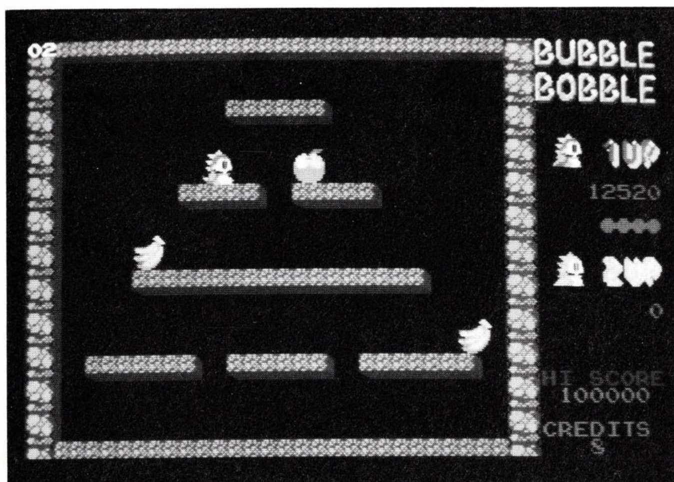
Kampfgeblubber

C64-Besitzer kennen das blasenspuckende Drachengespann Bub und Bob bereits. Schon am „Brotkasten" und in den Spielhallen wurden Freaks massenhaft „Bubble Bobble"-süchtig. Dieses Schicksal soll nun auch Amiga-Spieler ereilen. Kein Problem! Denn wer legt freiwillig den Joystick aus der Hand, wenn 100 Level voller Minimonster und unzähliger Bonusgegenstände auf ihn warten?

Zwei Drachen eilen über die Plattformen und hüpfen von Ebene zu Ebene. Überall werden sie von kleinen Monstern verfolgt und überrannt. Doch sie sind nicht wehrlos. Sie retten sich, indem sie alle Bösewichter mit großen Seifenblasen vollspucken und darin einschließen. Wenn sie — hops — dagegenspringen, verwandeln sich die Blasen in köstliches Obst, leckere Bonbons, glitzernde Diamanten oder andere Bonusgegenstände. Schneli aufsammeln! Denn jeder Edelstein, jede Leckerei bringt Punkte.

Sobald das Drachenduo einen Level von Monstern gesäubert und Boni gehortet hat, reist es in den nächsten Level. In höheren Spielstufen sind die Plattformen komplizierter angeordnet, neue Ungeheuer tauchen auf. Sogar neue Waffen gibt es: Zum Beispiel einen Kettenschlitten, mit dem Bub und Bob durch den Screen rattern, um alle Bonusgegenstände und Monster mitzureißen. Wenn einer der niedlichen Drachen seine Leben verwirkt hat, muß sein grüner Kumpel die Monster allein vollspucken. Grafisch ist „Bubble Bobble" nicht spektakulär. Allerdings lebt das Spiel auch nicht von der Grafik, sondern von dem süchtigmachenden Spielprinzip. Zudem ist das Programm sehr nett realisiert worden; niedliche Spielfiguren tummeln sich, es wird nicht mit kriegerischen Waffen gekämpft, sondern gebubbelt.

(cb)



Klein-Dino verwandelt den Kampfplatz in einen Obstladen.

Name: Bubble Bobble	
Preis: 65 Mark	
Hersteller: Firebird	
Spaß	10
Grafik	7
Sound	7
Idee	9
Schwierigkeit	7

Flugsimulant mit destruktivem Touch



Über den Wolken fasziniert die Illusion endloser Freiheit.

Lufträume erobern, Destruktionstriebe ausleben, manuelles Geschick beweisen, Flugsimulation „Jet" erfüllt jeden Wunsch.

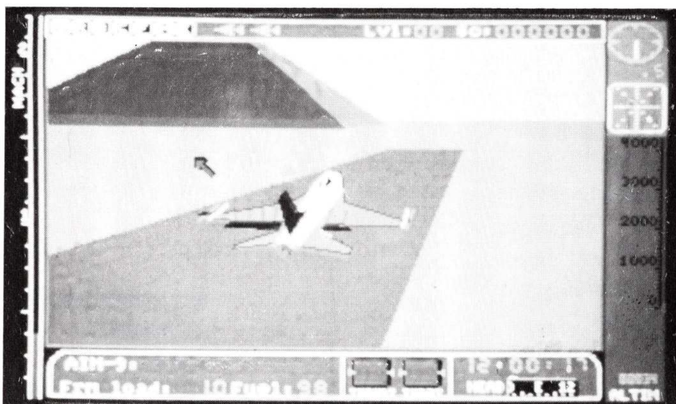
Vor über einem Jahr kündigte SubLogic ihren „Jet" mit großem Werberummel an. Dann wurde das lang erwartete Programm gestrichen. Erstaunlich, aber wahr: „Jet", Nachfolger des legendären „Flight Simulator II" ist dennoch erhältlich. „Jet" präsentiert sich als martialische Variation des Vorbilds. Flugbegeisterte fliegen entweder eine F16 oder starten mit einer F18 vom Flugzeugträger. Sofort fallen die beabsichtigten Übereinstimmungen mit dem Flight Simulator auf. Steuereinrichtungen, Fensterorganisation für Karten und Perspektive sind nahezu identisch. Die Zeichengeschwindigkeit ist genauso langsam wie beim Vorbild. So ist es bedenklich, ein weiteres Fenster zu öffnen, denn die Bildersequenz läßt merklich nach. Probieren wir es aus. Diskette rein und Flugzeug aussuchen. Es erscheint der gewählte Jet mit entsprechendem Startareal. Da es technisch unmöglich ist, mit einer F16 vom Flugzeugträger zu starten, ist dies auch hier nicht vorgesehen. Anzahl und Stärke der computergesteuerten Gegner bestimmen die Kampfmethode. Im einfachsten Modus sind Bodentruppen die einzigen potentiellen Ziele, während in der schwierigsten Stufe alles aufgeboten wird, was Bits und Bytes hergeben. Je nachdem, wer oder was als Zielscheibe dienen soll, muß der Jet richtig bewaffnet werden. Hier allerdings wird es unrealistisch. Für die F16 liegen Waffen im Arsenal, die üblicherweise der F18 vorbehalten sind. Am Instrumentenbrett fehlen typenbedingte Unterschiede. Grafische Mängel sind bei der Landschaftsgestaltung festzustellen. Landschaftliche Öde im Kampfgebiet wird nur von zwei Gebäudeanlagen unterbrochen, nämlich vom eigenen und vom gegnerischen Flughafen. Vielleicht läßt sich der Mangel beheben; schließlich lassen sich die Landschafts-Disketten vom Flight Simulator einladen. Auch hier Frustration. Zwar donnert man nun im Cockpit mit Mach 2 über die Golden Gate Brücke, doch taugt dieses weniger monumentale als immaterielle Stahlskelett weder als Ziel noch als Hindernis. Zahlreiche Features machen „Jet" zwar zum überdurchschnittlichen Ballerspiel, jedoch weniger zur Flugsimulation. Lichtblick für den Modem-Jetset: Luftkämpfe können über Telefon ausgetragen werden.

(O. Röhrig)

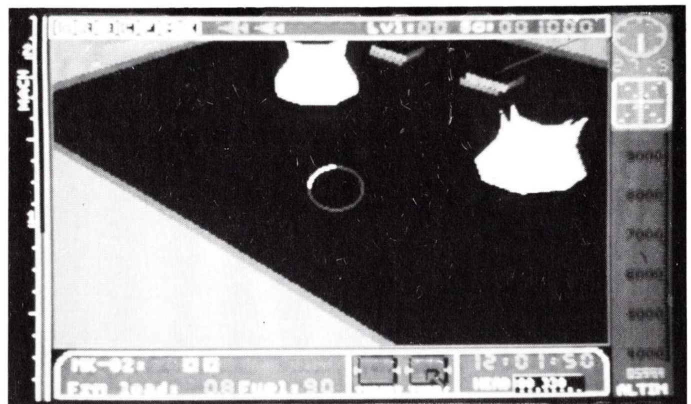
Name: Jet

Preis: 119 Mark
Hersteller: SubLogic

Spaß	7
Grafik	8
Sound	6
Idee	7
Schwierigkeit	7



„Jet" startet zwischen Pharaonengräbern . . .



. . . und donnert über Atomkraftwerke hinweg.

Impressum

Chefredakteur: Manfred S. Schmidt (mss) (verantwortlich, Anschrift siehe Redaktion)
Redaktion: Uwe Knierim (uk), Mike Nickles (mn), Ute Bahn (ub)
Redaktionsassistentin: Siggi Pöschel (sp)
Gestaltung: Karin Wirth
Disketten-Service: Wolfgang Kurtz (wok), Tel.: 0 89/22 47 20
Anschrift der Redaktion: CW Publikationen (Amiga-Welt), Postfach 40 04 29, Rheinstraße 28, 8000 München 40, Telefon: 0 89/3 60 86-0, Telex: 5 215 350 comw d, Telekopierer: 0 89/3 60 86-1 09
Auslandsredaktionen:
Österreich: COMPUTERWELT Österreich, CW-Publikationen Verlagsgesellschaft mbH, Zieglergasse 6/1, A-1070 Wien, Tel.: (00 43) 222/93 05 08
Schweiz: COMPUTERWORLD Schweiz, Witikonerstr. 15, Postfach 253, CH-8030 Zürich, Tel.: (01)55 10 77, Telex: 816 710 cw cich
USA: CW-COMMUNICATIONS INC., 375 Cochituate Road, Box 9171, USA-Framingham, Mass. 01701, Tel.: 001/617/879 07 00, Tx.: 00230/951 153 computwrld fmh
Japan: COMPUTERWORLD/Japan, 1- 19-7, Shintomi Chuoku, J-Tokyo 104, Tel.: 00 81/3/5 51 38 82
China: China COMPUTERWORLD, 74 LuGuGun Road, P.O. Box 750, RC-Beijing 100039, Tel.: 00 88/6/814-61 74
Verlagsrepräsentanten:
Großbritannien: Beere Hobson Ass., Euan C. Rose, 34 Warwick Road, Kenilworth, GB-Warwickshire CV8 1HE, Tel.: 00 44/926/51 24 24.
USA: CW International Marketing Services, Diana La Muraglia, 375 Cochituate Road, Box 880, USA-Framingham, Mass. 01701, Tel.: 001/617/879 07 00, Tx.: 00230/951 153 computwrld fmh
Taiwan: Alice Chen ACE Media, Agency Co., Ltd., P.O.Box 26-578, Taipei R.O.C., Tel. 021721-4302, Telex 14142 acegroup
Japan: Steven Yamada, Tokyo Representative Corp., Sanshin Kogyo Jimbocho 3F, Chiyoda-ku, Tokyo 101 Japan, Tel.: 230-4117, Tlx.: (781) J26860

Sonderheft-Bestellungen: Direkt beim Verlag, Abt. Vertrieb (Anschrift s. u.) gegen Verrechnungsscheck + DM 2,– Porto oder per Nachnahme sowie über den Buchhandel

ISSN 0933-8616

Vertrieb Handelsauflage: MZV Moderner Zeitschriften Vertrieb GmbH, Breslauer Straße 5, 8057 Eching, Tel.: 089/31 90 06-0, Telex: 522 656.
Telefon-Durchwahl: 089/3 60 86 ( )
Vertriebsleitung: Brigitte Schleibinger (-155/-156)
Anzeigenleitung: Sylvia Stier (-161); (verantwortlich für Anzeigen, Anschrift siehe unter Anzeigen)
Anzeigenverkaufsleitung: Barbara Schönberger (-130)
Anzeigendisposition: Ursel Sauter (-126)

Druck und Beilagen:
Carl Gerber Grafische Betriebe, Muthmannstraße 4, 8000 München 45, Tel.: 089/3 23 93-233 (Anschrift für Beilagen)
Anschrift für Anzeigen und Vertrieb: Postfach 400 429, Rheinstraße 28, 8000 München 40, Telefon: 0 89/3 60 86-0, Telex: 5 215 350 comw d, Telekopierer 089/3 60 86-1 09
Disposition:
Rainer Oberländer (Leitung), Stefan Liba
Zahlungsmöglichkeiten:
Bayerische Vereinsbank, BLZ 700 202 70, Konto-Nr. 116 000, Pschk. München 97 40-800
Erfüllungsort, Gerichtsstand: München
Verlag: CW-Publikationen Verlagsgesellschaft mbH, Rheinstraße 28, 8000 München 40, Telefon: 089/3 60 86- 0, Telex: 5 215 350 comw d, Telekopierer: 089/3 60 86-109
Art Director: Darinka Bratuša
Produktionsleitung: Heinz Zimmermann
Redaktionsdirektor: Dieter Eckbauer
Kaufm. Leitung: Jürgen T. Reischl
Geschäftsführer: Eckhard Utpadel, Walter Boyd, Patrick McGovern

INSERENTENVERZEICHNIS

Kleinanzeigen-Service

AMIGA WELT

Die Zeitschrift für den Amiga Anwender

Jede weitere Zeile nur DM 1,— Bitte deutlich schreiben. Zwischen den Worten jeweils ein Kästchen freilassen.

Ob 1, 2 oder 6 Zeilen — jede Kleinanzeige für nur DM 5,—

Achtung! Sie erhalten keine Rechnung. Chiffre-Anzeigen sind nicht möglich. Damit Ihre Anzeige veröffentlicht wird, legen Sie bitte den entsprechenden Betrag in bar oder als Verrechnungsscheck bei.

Gewünschte Rubrik
☐ Verkaufe
☐ Suche ☐ Tausche

Produktbereich
☐ Hardware ☐ Peripherie ☐ Sonstiges
☐ Software ☐ Disketten

Bitte ankreuzen!

CW-Publikationen
AmigaWelt — Anzeigenverkauf
Rheinstr. 28

D-8000 München 40

Absender

Name Vorname

Straße, Hausnr./Postfach

PLZ Ort

Vorwahl Telefon

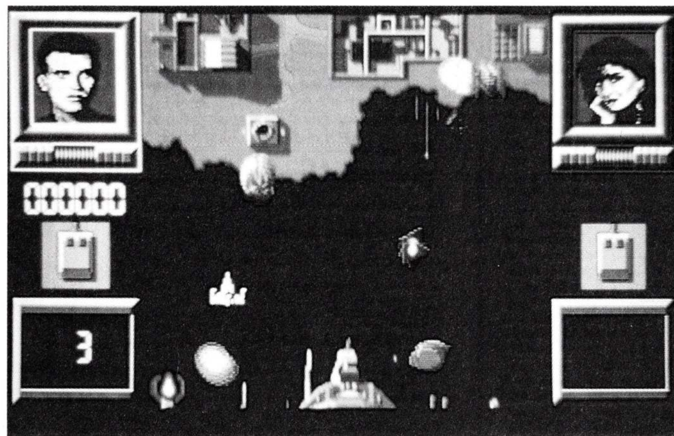
Datum Unterschrift (bei Minderjährigen Unterschrift d. ges. Vertreters)

Der Betrag von DM ______ liegt
☐ in bar/
☐ als Verrechnungsscheck bei
(Zutreffendes bitte ankreuzen).

Ausgefüllten Bestellschein in ein Fensterkuvert stecken und gleich abschicken!

Schatten im Orbit

„Black Shadow“, ein bewohnter Asteroid, ist in die Umlaufbahn um unsere Mutter Erde eingedrungen. Was haben die galaktischen Unwesen vor? Man rechnet mit dem Schlimmsten und schickt unerschütterliche Kampfpiloten in den Orbit. Über drei gefahrvolle Sektoren donnern ihre martialischen Flieger, über Militärbasen, industrielle Anlagen zum Solar-Hyperreaktor, dem Kernstück des Himmelskörpers. Gelingt es, dieses Energiezentrum zu zerstören, explodiert der unheimliche Himmelskörper und die Gefahr ist gebannt.



Kunterbunte Aliens bedrohen den irdischen Kampfflieger.

Also, Freiwillige vor! Vor dem Einsatz gibt man fix die Steuereinheit ein — soll mit Joystick oder Maus zerschmettert und zerschossen werden? Jeder Spieler sucht sich unter den Damen und Herren einen Piloten aus. Wer allein ist, muß nicht auf Mitspieler warten, denn auch Einzelkämpfer können in den Orbit starten. Und schon im nächsten Moment sieht man seinen Flieger auf dem Bildschirm. Er droht mit Luftbomben zu kollidieren. Rotierende Scheiben greifen an. Dreieckige Raumschiffe und farbige Kugeln stürzen sich wie Kamikaze-Flieger auf den irdischen Retter. Er verfügt über sechs Bildschirmleben. Aber wer glaubt, deshalb unaufmerksam sein zu dürfen, wird den tückischen Feinden schnell erliegen. Also schießt er auf alles, was sich regt oder weicht aus. Maustaste — oder Feuerknopf — kommen nicht zur Ruhe, während der Spieler bunte Sprites zerschmettert. Im Hintergrund erstreckt sich eine detailliert dargestellte Asteroidenlandschaft. Die Grafik ist über jede Kritik erhaben. Genauso lobenswert ist der Sound: Bombastische Titelmusik und Spielgeräusche hört der Spieler in Stereo.

(cb)

Name: Black Shadow
Preis: 70 Mark
Hersteller: CRL

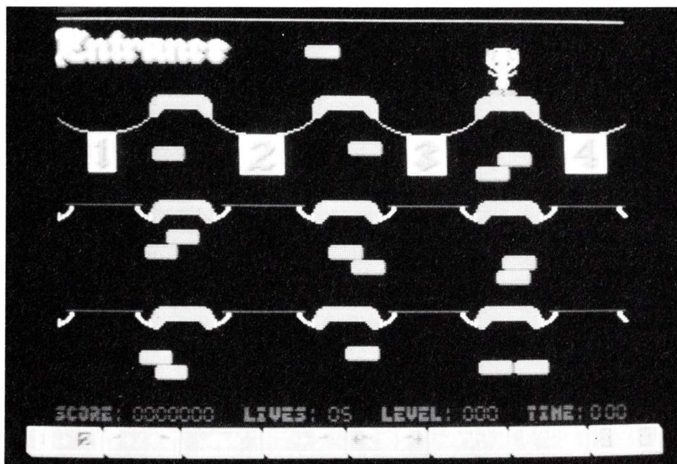
Spaß	8
Grafik	8
Sound	8
Idee	6
Schwierigkeit	9

Kalte Ente!

Larrie, entengesichtiger Antiheld und Abenteurer, hat sich einen ziemlich lästigen Job eingehandelt. Er soll äußerst befremdliche Eindringlinge aus einer ehemals friedlichen Stadt vertreiben. Es handelt sich dabei um besonders bösartige Kreaturen, nämlich die Gattung der Ardies. Ein Kampf zwischen Gut und Böse entbrennt. Larrie und die Ardies toben über, unter und zwischen Plattformen. Larrie eilt über Ebenen, sammelt Bonusfigürchen auf und turnt über Trampoline von einer Etage zur anderen. Doch sein Job beschränkt sich nicht auf sportliche Intermezzi und das Ausleben seiner Sammelleidenschaft.

Ständig hecheln die gefährlichen Ardies hinter Enten-Larrie her und versuchen, eines seiner Bildschirmleben zu beenden, indem sie ihn berühren. Doch unser Antiheld ist nicht wehrlos. Mitunter killt er seine dreisten Widersacher oder rennt hurtig davon. Für den Spieler ist es allerdings besonders spaßig, die Ardies zu erschrecken. Aber erstmal muß man sie überhaupt zu sehen bekommen! Hier hört das Amüsement auf. Wer ungeduldig ist und nicht gern tüftelt, wirft an diesem Punkt das Handtuch. Während des Tests tauchte im ersten Level — in der Entrance Hall — weit und breit kein Ardie auf; Gegenstände gab's auch nicht. Mit Trick 17 gelang es endlich, in den zweiten Level vorzustoßen: Man spielt im Zwei-Spieler-Modus und läßt den Larrie des ersten Spielers alle Leben verlieren. Automatisch landet man dann im zweiten Level. Seltsam! Doch das ist nicht der einzige technische Mangel. Sollte jemand während des Joystick-Spiels auf einen Mausknopf drücken, stürzt das Spiel ab. Wer den Finger zu lange auf den Feuerknopf preßt, sieht seinen Larrie in der Luft hängen. Diese Macken kann selbst Carsten Obarskis Spitzensound nicht ausgleichen.

(cb)



Larrie bewegt sich unentenhaft über die Plattformen.

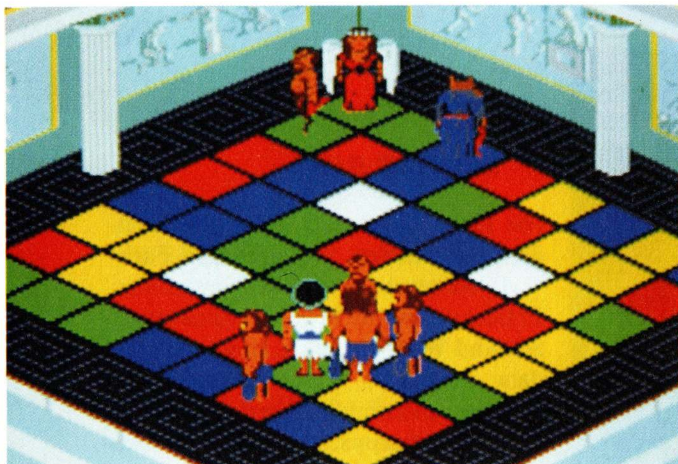
Name: Larrie and the Robbery of the Ardies
Preis: 49,95 Mark
Hersteller: Crusus

Spaß	2
Grafik	5
Sound	9
Idee	7
Schwierigkeit	8

Göttliche Quizrunde

Jetzt spielen die Götter „Trivial Pursuit"! Natürlich ist Zeus schuld. Er will wissen, welcher seiner Untergötter der Weiseste und Mächtigste ist. Hermes, Hecate und Aphrodite werden in den Tempel des Apollo zum Gesellschaftsspiel gebeten. Der Hausherr des heiligen Gemäuers muß ebenfalls mitraten. Dort, auf den Höhen des Olymp und in den Katakomben des Hades, tragen sie ihren Wettkampf aus, das heißt, nicht sie selbst. Götter sind da vornehm und schicken ihre halbgöttlichen Verwandten und Krieger an den Start. Das schont die Nerven und vermeidet Verletzungen am göttlichen Körper. So kann den vier Spielern, die mit göttergleichem Geschick an Tastatur, Maus oder Joystick wirken, nichts passieren. Sie stellen siegesverdächtige Kampfeinheiten zusammen und ziehen gegen den omnipotenten Zeus in's Spielfeld. Jede richtige Antwort bringt den göttlichen Stellvertreter ein Kästchen weiter. Gleichzeitig werden Weisheitspunkte gescheffelt. Wer viele Punkte auf dem Konto hat, kann den Gegnern stärkere Krieger entgegenstellen. Aber der Weg zum Lorbeer ist steinig; universelle Fragen bringen die überirdischen Hirne gehörig auf Trab. Begegnen sich zwei Spieler auf einem Kästchen, müssen sie einander herausfordern. Entscheidung bringt eine Schnellraterunde zu frei wählbaren Themen. Der Schwierigkeitsgrad der Fragen richtet sich nach der Qualität des stärkeren Kontrahenten. Sollten selbst anspruchsvolle Fragen irgendwann langweilen, kann sich jeder Spielgott Fragen ausdenken, an denen sich seine Mitgötter die Zähne ausbeißen.

(J. Döllner)



Frageolympiade: Halbgötter als Spielsteine.

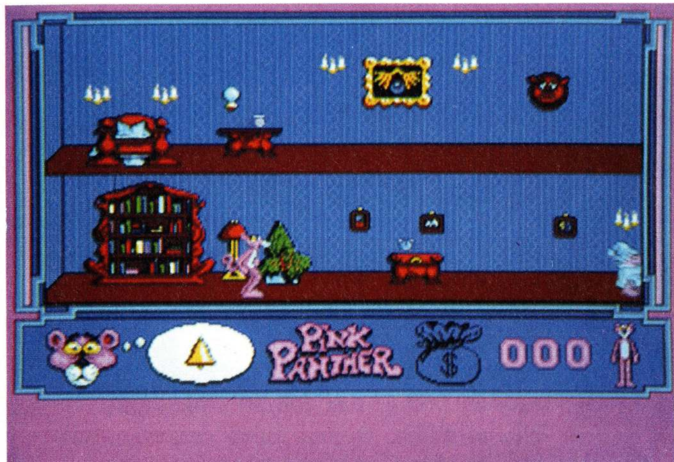
Name: Powerplay, The Game of the Gods
Preis: 65 Mark
Hersteller: Arcana

Spaß	8
Grafik	4
Sound	3
Idee	5
Schwierigkeit	6

Rosaroter Langfinger

Paulchen Panther hat das hektische Stadtleben satt. Er träumt davon, unter Palmen zu dösen. Doch so eine Fernreise kostet Geld, eine Menge Geld! Welch ein deprimierender Gedanke, die benötigte Summe mit ehrlicher Arbeit aufzubringen. Paulchen beschließt, sich tagsüber als Hausdiener bei reichen Leuten zu verdingen und nachts die Urlaubskasse zu füllen. Betuchte Herren haben hohe Ansprüche. Also kratzt der Butler in spe seine letzten Groschen zusammen und ersteht einen Zylinderhut. Im Hause seines ersten Arbeitgebers ließen sich nachts Wertsachen organisieren, wäre da nicht ein kleines Problem; der Hausherr ist nämlich somnambul. Und sobald er aus seinen Schlafwandeltouren erwacht und die kriminellen Umtriebe seines rosaroten Butlers entdeckt, ruft er den berühmten Inspektor Clouseau. Nichts fürchtet Paulchen mehr als diesen wackeren Polizeiinspektor. Denn der würde dem Traum vom Inselurlaub ein jähes Ende setzen. Da greift der gerissene Panther schnell in die Trickkiste und benutzt kuriose Hilfsmittel, um den hausherrlichen Schlafwandler am Erwachen zu hindern. Er steuert seinen Boss mit einer Zwille, kleinen Glöckchen oder einem aufblasbaren Plastikpaul um alles herum, was ihn aufwekken könnte. Und kommt tatsächlich der wackere Inspektor in's Haus, läßt Paulchen ihn in einem aufblasbaren Loch verschwinden. Wenn alle Wertgegenstände den Besitzer gewechselt haben, tritt der jetzt noch besser ausgerüstete Panther den nächsten Job an. Fünf reiche Schlafwandler muß der urlaubsreife Butler plündern, um seinen Inseltraum realisieren zu können. Spieler dürfen fleißig beim Stehlen helfen und sich währenddessen über feine Grafik und Animation freuen. Der Sound ist weniger erfreulich.

(Salus)



Auf leisen Pfoten und mit langen Fingern sammelt Paul für die Urlaubskasse. Sein Boss schlafwandelt derweil.

Name: The Pink Panther
Preis: 60 Mark
Hersteller: Magic Bytes

Spaß	7
Grafik	6
Sound	5
Idee	7
Schwierigkeit	9

VORSCHAU

Juli/August



Komponieren, Soundsampling, Digitizing, vier Stereokanäle: Schlagworte, mit denen sich die audiophilen Fähigkeiten des Amiga umreißen lassen. Amiga Welt erläutert Programmtechnisches, vermittelt Grundlagen und berichtet über Anwendungen — zum Beispiel in der Unterhaltungsbranche. Erscheinungstermin: 22. 8. 1988



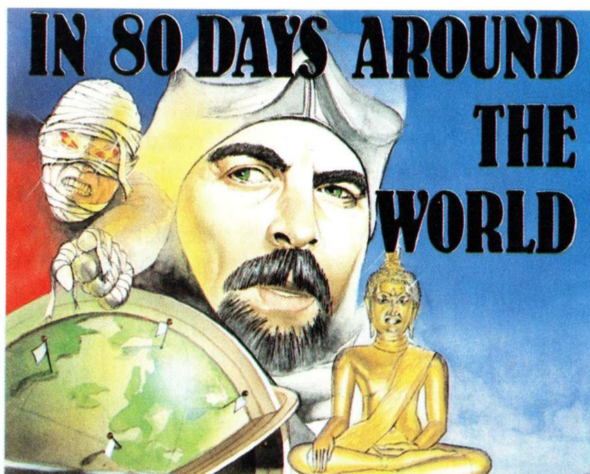
Blende auf

Nie mehr Über- oder Unterbelichtungen, Unschärfen, Schlieren und Farbverfälschungen bei Bildschirmfotos. Erfahrungen mit Blende, Belichtungszeit und Wahl des Films lassen sich mit dem in BASIC geschriebenen Programm „Testbild“ auswerten. Gelungene Schnappschüsse illustrer Bildschirminhalte gelingen nun jedem.



Planeten im VideoRAM

Voyager 2 schickt Daten aus der Tiefe des Universums zur Erde. Doch Daten vermitteln keine Atmosphäre. Joel Hagen verarbeitete die abstrakten Informationen zu realistischen Ausblikken auf unirdische Landschaften und Horizonte. Amiga und DeluxePaint II machten es möglich.

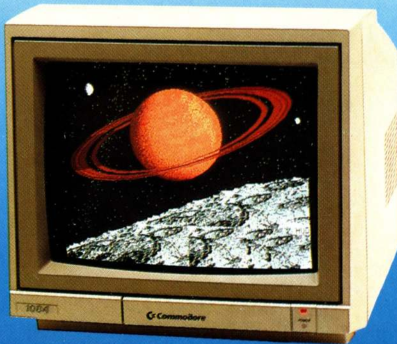


Die Wette gilt!

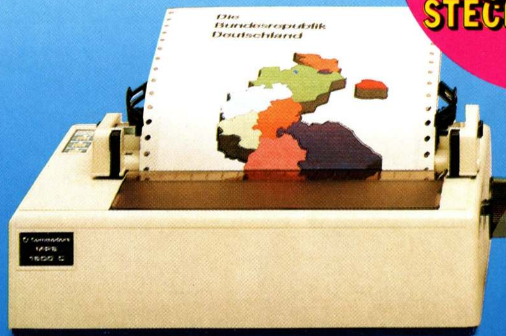
„In 80 Tagen um die Welt“ mag heute kein Problem sein. Dem Phileas Fogg, einer Romanfigur des 19. Jahrhunderts, hätte der tollkühne Rekordversuch beinahe das ganze Vermögen gekostet. Daß er das damals Unglaubliche schaffte, wissen Jules-Verne-Leser. In der Computer-Abenteuerreise von heute entscheidet sich der Spieler für Ruhm und gegen Ruin.

EINSTECKEN UND VOLL LOSLEGEN
AMIGA 500 ORIGINALZUBEHÖR





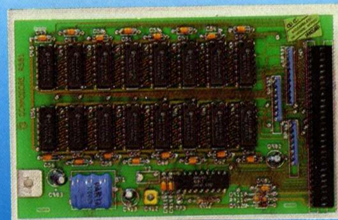
Farbmonitor A 1084
Das Schärfste für Ihren Amiga.
2.000 Zeichen. 14" Bildschirm. Entspiegelt.



Farbdrucker MPS 1500 C
Da geht's bunt zu. Auf Wunsch auch schwarz.
120 Zeichen/Sek. Normal- und Schönschrift.



A 520 Video-Adapter
Für alle, die TV oder Videomonitor an den A 500 anschließen wollen.



Speichererweiterung A 501. Heavy tuning mit clockset! Erweitert RAM um 512 KB auf 1 MB.





A 1010 Disk Drive
Damit geht's rund. Mit 3,5",
Speicherkapazität 880 KB.



Amiga 500
Klasse Software – Riesenauswahl!
Von wem und wie?
Steht alles im Amiga Katalog. Sofort beim Händler besorgen!

Wenn Sie ohne Fehlstarts sofort loslegen wollen,
verwenden Sie nur Original-Peripherie von Commodore.
Damit läuft Ihr Amiga 500 zur Höchstform auf.
Einstecken – und schon geht die Post ab. Denn da paßt
jedes Teil zum anderen.
Commodore – Marktführer bei Mikrocomputern.



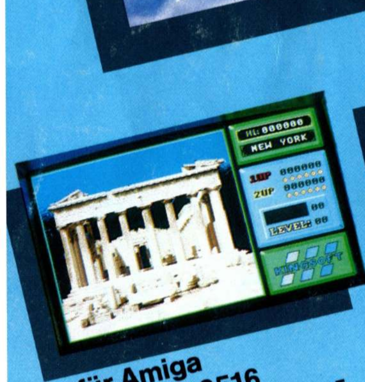


Stop! Topaktuell – KINGSOFT! Neue Computerspiele soeben eingetroffen

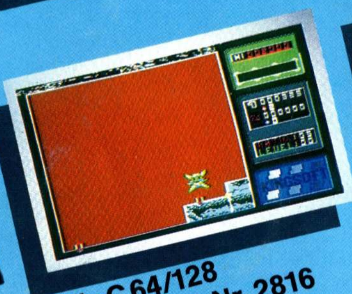
Maniax
Das empfehlenswerte Spiel zum Superpreis!

Alarm in der Großstadt!
Ein undurchdringbarer Nebel hat sich über die Häuser gelegt und nimmt einem jede Sicht. Nun ist man an Sie herangetreten, diesen Zustand zu beenden. Mit Ihrem Maniax können Sie Stück für Stück wieder Licht in die Stadt bringen.

„Maniax spielt sich ausgesprochen frisch – den zahlreichen Extras und den schönen Grafiken sei Dank." (Power Play)



für Amiga
Best.-Nr. 8516
nur DM 49.95



Für C 64/128
Disk. Best.-Nr. 2816
Kass. Best.-Nr. 2516
nur DM 29.95

für C16 (64K), Plus/4
Disk. Best.-Nr. 3816
Kass. Best.-Nr. 3516
nur DM 29.95

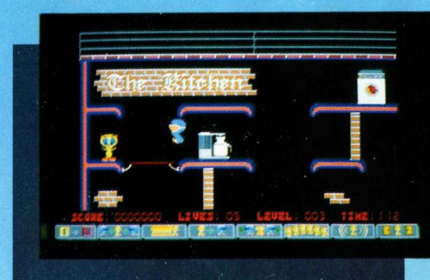


für IBM PC
Disk. Best.-Nr. 9516
nur DM 49.95



für Atari ST
Best.-Nr. 7516
nur DM 49.95

Larrie für Amiga
Dieses Spiel müssen Sie haben!



Larrie, der kleine Vogel springt von Trampolin zu Trampolin. Aber es droht Gefahr – die Ardies kommen.
Best.-Nr. 8535 nur DM 49.95

Iridon für Amiga
Der brandheiße Action-Renner!



Superschnelles horizontal scrollendes Spiel. Mit prächtiger Grafik und vielen Digisounds. Der gesamte PAL-Bildschirm wird als Spielfläche ausgenutzt.
Best.-Nr. 8528 nur DM 49.95

Telefonische Bestellung 4

Natürlich haben sie auch die Möglichkeit, Kingsoft-Spiele telefonisch zu bestellen.
02 41/15 20 51

Diesen Bestellschein einfach ausschneiden und auf eine Postkarte kleben.
Absender nicht vergessen.
Postkarte mit 60 Pfennig frankieren. 4

Anzahl	Bestell-Nr.	Name des Spiels	Preis

Zahlung per Nachnahme zzg. Versandkosten

mein Computertyp

☐ Bitte ankreuzen, wenn Sie in Zukunft kostenlos unsere aktuelle Information erhalten möchten

Kingsoft GmbH · Grüner Weg 29 · 5100 Aachen



KINGSOFT GmbH · Grüner Weg 29
D-5100 Aachen · Tel.: 02 41/15 20 51